

FFFFFFFFFFFFFFFF	111	111	XXX	XXX
FFFFFFFFFFFFFFFF	111	111	XXX	XXX
FFFFFFFFFFFFFFFF	111	111	XXX	XXX
FFF	111111	111111	XXX	XXX
FFF	111111	111111	XXX	XXX
FFF	111111	111111	XXX	XXX
FFF	111	111		
FFF	111	111	XXX	XXX
FFF	111	111	XXX	XXX
FFFFFFFFF.FFF	111	111		
FFFFFFFFFFFFFFFF	111	111	XXX	
FFFFFFFFFFFFFFFF	111	111	XXX	
FFF	111	111		
FFF	111	111	XXX	XXX
FFF	111	111	XXX	XXX
FFF	111	111	XXX	XXX
FFF	111	111		
FFF	111	111	XXX	XXX
FFF	111	111	XXX	XXX
FFF	1111111111	1111111111	XXX	XXX
FFF	1111111111	1111111111	XXX	XXX
FFF	1111111111	1111111111	XXX	XXX

CCCCCCCC	RRRRRRRR	EEEEEEEEEE	AAAAAA	TTTTTTTTTT	EEEEEEEEEE	
CCCCCCCC	RRRRRRRR	EEEEEEEEEE	AAAAAA	TTTTTTTTTT	EEEEEEEEEE	
CC	RR	RR	AA	TT	EE	
CC	RR	RR	AA	TT	EE	
CC	RR	RR	AA	TT	EE	
CC	RR	RR	AA	TT	EE	
CC	RRRRRRRR	EEEEEEEEEE	AA	TT	EEEEEEEEEE	
CC	RRRRRRRR	EEEEEEEEEE	AA	TT	EEEEEEEEEE	
CC	RR	RR	AAAAAAAAAA	TT	EE	
CC	RR	RR	AAAAAAAAAA	TT	EE	
CC	RR	RR	AA	TT	EE	
CC	RR	RR	AA	TT	EE	
CC	RR	RR	AA	TT	EE	
CC	RR	RR	AA	TT	EE	
CCCCCCCC	RR	EEEEEEEEEE	AA	TT	EEEEEEEEEE	....
CCCCCCCC	RR	EEEEEEEEEE	AA	TT	EEEEEEEEEE	....

```

LL               IIIII
LL               IIIII
LL               II
LL               II
LL               II
LL               II
LL               II
LL               II
LL               II
LL               II
LL               II
LL               II
LL               II
LL               II
LL               II
LLLLLLLLLLLLLL  IIIII
LLLLLLLLLLLLLL  IIIII

SSSSSSSS
SSSSSSSS
SS
SS
SS
SS
SSSSSS
SSSSSS
SS
SS
SS
SS
SSSSSSSS
SSSSSSSS

```



CREATE

L 15  
16-Sep-1984 00:06:06  
14-Sep-1984 12:30:13VAX-11 Bliss-32 V4.0-742  
DISK\$VMSMASTER:[F11X.SRC]CREATE.B32;2  
Page 1  
(1)CR  
VO

```
0001 0 MODULE CREATE (
0002 0      LANGUAGE (BLISS32),
0003 0      IDENT = 'V04-001'
0004 0      ) =
0005 1 BEGIN
0006 1
0007 1
0008 1 *****
0009 1 *
0010 1 *  COPYRIGHT (c) 1978, 1980, 1982, 1984 BY
0011 1 *  DIGITAL EQUIPMENT CORPORATION, MAYNARD, MASSACHUSETTS.
0012 1 *  ALL RIGHTS RESERVED.
0013 1 *
0014 1 *  THIS SOFTWARE IS FURNISHED UNDER A LICENSE AND MAY BE USED AND COPIED
0015 1 *  ONLY IN ACCORDANCE WITH THE TERMS OF SUCH LICENSE AND WITH THE
0016 1 *  INCLUSION OF THE ABOVE COPYRIGHT NOTICE. THIS SOFTWARE OR ANY OTHER
0017 1 *  COPIES THEREOF MAY NOT BE PROVIDED OR OTHERWISE MADE AVAILABLE TO ANY
0018 1 *  OTHER PERSON. NO TITLE TO AND OWNERSHIP OF THE SOFTWARE IS HEREBY
0019 1 *  TRANSFERRED.
0020 1 *
0021 1 *  THE INFORMATION IN THIS SOFTWARE IS SUBJECT TO CHANGE WITHOUT NOTICE
0022 1 *  AND SHOULD NOT BE CONSTRUED AS A COMMITMENT BY DIGITAL EQUIPMENT
0023 1 *  CORPORATION.
0024 1 *
0025 1 *  DIGITAL ASSUMES NO RESPONSIBILITY FOR THE USE OR RELIABILITY OF ITS
0026 1 *  SOFTWARE ON EQUIPMENT WHICH IS NOT SUPPLIED BY DIGITAL.
0027 1 *
0028 1 *
0029 1 *****
0030 1
0031 1 ++
0032 1
0033 1 FACILITY: F11ACP Structure Level 2
0034 1
0035 1 ABSTRACT:
0036 1
0037 1      This module processes the create function. It creates a file with the
0038 1      attributes requested, enters it in a directory if desired, and
0039 1      accesses it if requested.
0040 1
0041 1 ENVIRONMENT:
0042 1
0043 1      STARLET operating system, including privileged system services
0044 1      and internal exec routines.
0045 1
0046 1 --
0047 1
0048 1
0049 1 AUTHOR: Andrew C. Goldstein, CREATION DATE: 28-Mar-1977 15:05
0050 1
0051 1 MODIFIED BY:
0052 1
0053 1      V04-001 CDS0006      Christian D. Saether      12-Sep-1984
0054 1      Modify test for re-reading file header after ENTER
0055 1      (CDS0004).
0056 1
0057 1      V03-042 CDS0005      Christian D. Saether      31-Aug-1984
```

CREATE  
V04-001

M 15

16-Sep-1984 00:06:06

14-Sep-1984 12:30:13

VAX-11 Bliss-32 V4.0-742

DISK\$VMSMASTER:[F11X.SRC]CREATE.B32;2

Page 2  
(1)

```

58      0058 1
59      0059 1
60      0060 1
61      0061 1
62      0062 1
63      0063 1
64      0064 1
65      0065 1
66      0066 1
67      0067 1
68      0068 1
69      0069 1
70      0070 1
71      0071 1
72      0072 1
73      0073 1
74      0074 1
75      0075 1
76      0076 1
77      0077 1
78      0078 1
79      0079 1
80      0080 1
81      0081 1
82      0082 1
83      0083 1
84      0084 1
85      0085 1
86      0086 1
87      0087 1
88      0088 1
89      0089 1
90      0090 1
91      0091 1
92      0092 1
93      0093 1
94      0094 1
95      0095 1
96      0096 1
97      0097 1
98      0098 1
99      0099 1
100     0100 1
101     0101 1
102     0102 1
103     0103 1
104     0104 1
105     0105 1
106     0106 1
107     0107 1
108     0108 1
109     0109 1
110     0110 1
111     0111 1
112     0112 1
113     0113 1
114     0114 1

Defer building of ACL's until after initial extend
takes place so that the map pointer for a contiguous
file is in the primary header.

V03-041 CDS0004      Christian D. Saether      30-Aug-1984
Reread newly created header after ENTER because
it may have been flushed from the cache by a multi
header directory file.

V03-040 CDS0013      Christian D. Saether      14-Aug-1984
Modify creation of extension fcb chain, if necessary.

V03-039 LMP0298      L. Mark Pilant,          7-Aug-1984  16:22
Add the necessary protection checks for create-if.

V03-038 ACG0438      Andrew C. Goldstein,     1-Aug-1984  21:23
Fix link truncation error; release any existing
serialization lock before starting create

V03-037 LMP0288      L. Mark Pilant,          29-Jul-1984  13:56
Make sure that the ACL queue head of the new file is properly
initialized when copying the ACL from a prior version (this
bug introduced in LMP0284.)

V03-036 LMP0284      L. Mark Pilant,          26-Jul-1984  12:14
Fix call to ACL_INIT_QUEUE, since it was moved to ACLSUBR.

V03-035 ACG0440      Andrew C. Goldstein,     25-Jul-1984  14:27
Move setup of default access ACE to after attributes are written

V03-034 LMP0275      L. Mark Pilant,          23-Jul-1984  14:40
Don't try to propagate an ACL if there isn't one.

V03-033 ACG0437      Andrew C. Goldstein,     13-Jul-1984  15:27
Corrections to alternate file ownership: fix interface to
CHANGE_OWNER so that next version propagation works and
so that space charging is done correctly. Also add an
ACL entry for the creator to guarantee access.

V03-032 CDS0012      Christian D. Saether     29-Jun-1984
Add another call to read_header after copying info
in propagate_attr because primary header may have
been flushed from the cache.

V03-031 CDS0011      Christian D. Saether     22-Apr-1984
Modify access arbitration.

V03-030 CDS0010      Christian D. Saether     11-Apr-1984
Remove call to allocation_unlock after create_header
call because that routine does it now.

V03-029 CDS0009      Christian D. Saether     1-Apr-1984
Call ALLOCATION_UNLOCK prior to deleting previous file
version in supersede operations to eliminate possible
deadlock condition if the previous version is being
extended at the same time.
Also call ALLOCATION_UNLOCK after an ENTER because it
```



115	0115	1
116	0116	1
117	0117	1
118	0118	1
119	0119	1
120	0120	1
121	0121	1
122	0122	1
123	0123	1
124	0124	1
125	0125	1
126	0126	1
127	0127	1
128	0128	1
129	0129	1
130	0130	1
131	0131	1
132	0132	1
133	0133	1
134	0134	1
135	0135	1
136	0136	1
137	0137	1
138	0138	1
139	0139	1
140	0140	1
141	0141	1
142	0142	1
143	0143	1
144	0144	1
145	0145	1
146	0146	1
147	0147	1
148	0148	1
149	0149	1
150	0150	1
151	0151	1
152	0152	1
153	0153	1
154	0154	1
155	0155	1
156	0156	1
157	0157	1
158	0158	1
159	0159	1
160	0160	1
161	0161	1
162	0162	1
163	0163	1
164	0164	1
165	0165	1
166	0166	1
167	0167	1
168	0168	1
169	0169	1
170	0170	1
171	0171	1

may have extended the directory and thus be holding the allocation lock, also causing potential deadlock further on in a number of ways.

V03-028	ACG0412	Andrew C. Goldstein,	22-Mar-1984	18:19	Implement agent access mode support; add access mode to check protection call; make attribute propagation to self a NOP (when a file is entered as a new version of itself).
V03-027	ACG0408	Andrew C. Goldstein,	20-Mar-1984	17:54	Make APPLY_RVN and DEFAULT_RVN macros; Make rest of global storage based.
V03-026	ACG0405	Andrew C. Goldstein,	16-Mar-1984	15:12	Fix handling of file headers in CHANGE_OWNER
V03-025	CDS0008	Christian D. Saether	9-Mar-1984		Remember CURR_LCKINDX from primary context and set it in secondary after OPEN_FILE so that copy info has the right lock basis when writing acl's to the primary file's header.
V03-024	LMP0203	L. Mark Pilant,	29-Feb-1984	10:34	Add support for FIB\$V_PROPAGATE. This allow the propagation rules to apply on an enter operation as well as a create operation.
V03-023	LMP0189	L. Mark Pilant,	6-Feb-1984	13:54	Add support for FIB\$V_DIRACL. This allows the ACL of a directory file parent to be copied directly to the children (with the exception of NOPROPAGATE ACES).
V03-022	LMP0188	L. Mark Pilant,	3-Feb-1984	16:08	Add support for a classification block.
V03-021	CDS0007	Christian D. Saether	17-Jan-1984		Modify interface to DEFAULT_RVN.
V03-020	CDS0006	Christian D. Saether	27-Dec-1983		Use BIND_COMMON macro.
V03-019	LMP0174	L. Mark Pilant,	1-Dec-1983	14:01	Change routine name for default ACE propagation. Also, Add a call to a routine to do general propagation.
V03-018	CDS0005	Christian D. Saether	14-Sep-1983		Modify interface to SERIAL_FILE routine.
V03-017	ACG56916	Andrew C. Goldstein,	21-Jun-1983	18:25	Use central routine for date management
V03-016	LMP0156	L. Mark Pilant,	19-Sep-1983	15:43	Files not entered into a directory now get the process default protection.
V03-015	LMP0149	L. Mark Pilant,	13-Sep-1983	11:25	Correct a logic problem that caused problems during the

B  
C  
D  
E  
F  
G  
H  
I  
J  
K  
L  
M  
N  
B  
C  
D  
E  
F  
G  
H  
I  
J  
K  
L  
M  
N  
B  
C  
D  
E  
F  
G  
H  
I  
J  
K  
L  
M  
N  
B  
C  
D  
E  
F  
G  
H  
I  
J  
K  
L  
M  
N  
B  
C  
D  
E  
F  
G  
H  
I



172	0172	1	protection check of a write attribute operation.
173	0173	1	
174	0174	1	V03-014 LMP0148 L. Mark Pilant, 31-Aug-1983 13:29
175	0175	1	Make sure propagated attributes make it to the header.
176	0176	1	
177	0177	1	V03-013 CDS0004 Christian D. Saether 16-May-1983
178	0178	1	Release allocation lock after newly allocated file
179	0179	1	header is locked.
180	0180	1	
181	0181	1	V03-012 CDS0003 Christian D. Saether 4-May-1983
182	0182	1	Add call to SERIAL_FILE routine to interlock file
183	0183	1	processing.
184	0184	1	
185	0185	1	V03-011 CDS0002 Christian D. Saether 9-Apr-1983
186	0186	1	Reflect change to ACCESS_LOCK interface.
187	0187	1	
188	0188	1	V03-010 ACG0323 Andrew C. Goldstein, 25-Mar-1983 15:51
189	0189	1	Simplify backlink handling to track RENAME changes
190	0190	1	
191	0191	1	V03-009 ACG53759 Andrew C. Goldstein, 24-Mar-1983 15:10
192	0192	1	Update revision date & count & expiration on ENTER
193	0193	1	
194	0194	1	V03-008 LMP0091 L. Mark Pilant, 18-Mar-1983 16:14
195	0195	1	Add a condition handler to the attribute propagation to
196	0196	1	catch non-existent files. Also, copy the entire file name
197	0197	1	when creating a long file named file.
198	0198	1	
199	0199	1	V03-007 LMP0080 L. Mark Pilant, 14-Feb-1983 16:16
200	0200	1	Add a new routine that is called to propagate the attributes
201	0201	1	from either the previous version of the file or the parent
202	0202	1	directory as necessary.
203	0203	1	
204	0204	1	V03-006 ACG53050 Andrew C. Goldstein, 31-Jan-1983 13:59
205	0205	1	Remove RVN check from check for dummy file ID
206	0206	1	
207	0207	1	V03-005 CDS0001 Christian D. Saether 12-Jan-1983
208	0208	1	Call routine to take out file access lock.
209	0209	1	
210	0210	1	V03-004 LMP0059 L. Mark Pilant, 21-Dec-1982 11:17
211	0211	1	Always create an FCB when accessing a file header. This
212	0212	1	eliminates a lot of special casing in FCB handling.
213	0213	1	
214	0214	1	V03-003 LMP0047 L. Mark Pilant, 29-Sep-1982 12:05
215	0215	1	Put back in the volume protection check deleted by LMP0036.
216	0216	1	
217	0217	1	V03-002 LMP0036 L. Mark Pilant, 5-Aug-1982 13:50
218	0218	1	Shuffle the order that the protection checks are done to
219	0219	1	allow for ACL's.
220	0220	1	
221	0221	1	V03-001 LMP0016 L. Mark Pilant, 25-Mar-1982 13:18
222	0222	1	Remove diddling of the COMPLETE bit in the window segments.
223	0223	1	
224	0224	1	V02-021 ACG0265 Andrew C. Goldstein, 15-Feb-1982 9:50
225	0225	1	Fix order of expiration date handling
226	0226	1	
227	0227	1	V02-020 ACG0258 Andrew C. Goldstein, 26-Jan-1982 16:57
228	0228	1	Fix reference to RVN 1 in expiration date processing



CREATE  
V04-001

C 16  
16-Sep-1984 00:06:06  
14-Sep-1984 12:30:13

VAX-11 Bliss-32 V4.0-742  
DISK\$VMSMASTER:[F11X.SRC]CREATE.B32;2

Page 5  
(1)

```
: 229      0229 1 |
: 230      0230 1 | V02-019 ACG0230      Andrew C. Goldstein,      23-Dec-1981  22:59
: 231      0231 1 |      Add expiration date support
: 232      0232 1 |
: 233      0233 1 | V02-018 ACG0247      Andrew C. Goldstein,      23-Dec-1981  20:44
: 234      0234 1 |      Set revision date to creation date
: 235      0235 1 |
: 236      0236 1 | V02-017 ACG0245      Andrew C. Goldstein,      23-Dec-1981  20:40
: 237      0237 1 |      Don't write back link if file is a spool file
: 238      0238 1 |
: 239      0239 1 | V02-016 LMP0003      L. Mark Pilant,      8-Dec-1981  10:20
: 240      0240 1 |      Added byte limit quota check on window creation.
: 241      0241 1 |
: 242      0242 1 | V02-015 ACG0238      Andrew C. Goldstein,      11-Dec-1981  23:30
: 243      0243 1 |      Allow creation of dummy directory entries
: 244      0244 1 |
: 245      0245 1 | V02-014 ACG0208      Andrew C. Goldstein,      17-Nov-1981  15:16
: 246      0246 1 |      Add segmented directory reccrd support
: 247      0247 1 |
: 248      0248 1 | V02-013 ACG0167      Andrew C. Goldstein,      16-Apr-1980  19:25
: 249      0249 1 |      Previous revision history moved to F11B.REV
: 250      0250 1 | **
: 251      0251 1 |
: 252      0252 1 |
: 253      0253 1 | LIBRARY 'SYSS$LIBRARY:LIB.L32';
: 254      0254 1 | REQUIRE 'SRC$:FCPDEF.B32';
: 255      1245 1 |
: 256      1246 1 |
: 257      1247 1 | FORWARD ROUTINE
: 258      1248 1 | CREATE          : L_NORM,      ! CREATE function routine
: 259      1249 1 | PROPAGATE_ATTR  : L_NORM,      ! Propagate file attributes
: 260      1250 1 | PROPAGATE_HANDLER,      ! condition handler for above
: 261      1251 1 | COPY_INFO       : L_NORM;      ! Copy info from old to new file
```



```
263 1252 1 GLOBAL ROUTINE CREATE : L_NORM =
264 1253 1
265 1254 1 ++
266 1255 1
267 1256 1 FUNCTIONAL DESCRIPTION:
268 1257 1
269 1258 1 This routine processes the CREATE function. It creates a file with the
270 1259 1 attributes requested, enters it in a directory if desired, and
271 1260 1 accesses the file if requested.
272 1261 1
273 1262 1 CALLING SEQUENCE:
274 1263 1 CREATE ()
275 1264 1
276 1265 1 INPUT PARAMETERS:
277 1266 1 NONE
278 1267 1
279 1268 1 IMPLICIT INPUTS:
280 1269 1 CURRENT_VCB: VCB of volume
281 1270 1 IO_PACKET: packet of this I/O request
282 1271 1
283 1272 1 OUTPUT PARAMETERS:
284 1273 1 NONE
285 1274 1
286 1275 1 IMPLICIT OUTPUTS:
287 1276 1 PRIMARY_FCB: FCB of file if accessed
288 1277 1 CURRENT_WINDOW: window of file if accessed
289 1278 1 USER_STATUS: I/O status block of user
290 1279 1
291 1280 1 ROUTINE VALUE:
292 1281 1 1 if successful
293 1282 1 0 if error
294 1283 1
295 1284 1 SIDE EFFECTS:
296 1285 1 File created, blocks allocated, directory modified, file accessed, etc.
297 1286 1
298 1287 1 --
299 1288 1
300 1289 2 BEGIN
301 1290 2
302 1291 2 LITERAL
303 1292 2 ACE_LENGTH = $BYTEOFFSET (ACE$L_KEY) + 4;
304 1293 2
305 1294 2 LOCAL
306 1295 2 STATUS, | general return status
307 1296 2 K, | miscellaneous constant
308 1297 2 FCB_CREATED, | flag indicating new FCB created
309 1298 2 PACKET : REF BBLOCK, | address of I/O packet
310 1299 2 ABD : REF BBLOCKVECTOR [ABD$L_LENGTH], | buffer descriptors
311 1300 2 FIB : REF BBLOCK, | file identification block
312 1301 2 RESULT_LENGTH, | length of result string from ENTER
313 1302 2 RESULT : VECTOR [FILENAME_LENGTH+6, BYTE], | result string from ENTER
314 1303 2 LINK_DID : BBLOCK [FID$L_LENGTH], | header back link
315 1304 2 IDENT_AREA : REF BBLOCK, | pointer to file header ident area
316 1305 2 PCB : REF BBLOCK, | requestor PCB address
317 1306 2 ARB : REF BBLOCK, | access rights block of caller
318 1307 2
319 1308 2
```



```
: 320      1309 2      MAP_AREA      : REF BBLOCK,      ! file header map area
: 321      1310 2      IDX_FCB       : REF BBLOCK,      ! FCB of index file
: 322      1311 2      FCB          : REF BBLOCK,      ! FCB address
: 323      1312 2      UCB          : REF BBLOCK,      ! UCB pointer for RVN 1
: 324      1313 2      PRIMARY_VCB  : REF BBLOCK,      ! VCB of root volume
: 325      1314 2      HEADER       : REF BBLOCK,      ! address of file header
: 326      1315 2      NEW_HEADER   : REF BBLOCK,      ! Address of extension header
: 327      1316 2      ACL_CONTEXT,  : REF BBLOCK,      ! dummy ACL context longword
: 328      1317 2      ACE          : BBLOCK [ACE_LENGTH], ! buffer for ACE for file creator
: 329      1318 2      FUNCTION     : BLOCK [1];      ! function code qualifiers
: 330      1319 2
: 331      1320 2      EXTERNAL
: 332      1321 2      ACP$GB_WRITBACK : BITVECTOR ADDRESSING_MODE (ABSOLUTE),
: 333      1322 2      ! ACP write back cache enable
: 334      1323 2      SCH$GL_PCBVEC  : REF VECTOR ADDRESSING_MODE (ABSOLUTE), ! PCB vector
: 335      1324 2      EXE$GL_DYNAMIC_FLAGS : ADDRESSING_MODE (ABSOLUTE);
: 336      1325 2      ! Dynamic SYSGEN flags
: 337      1326 2
: 338      1327 2      EXTERNAL LITERAL
: 339      1328 2      EXE$V_CLASS_PROT;      ! Set if doing non-discretionary checks
: 340      1329 2
: 341      1330 2      BIND_COMMON;
: 342      1331 2
: 343      1332 2      EXTERNAL ROUTINE
: 344      1333 2      ACL_DELETEACL  : ADDRESSING_MODE (GENERAL), ! delete acls
: 345      1334 2      UPDATE_FCB    : L_NORM,      ! rebuild fcb from header
: 346      1335 2      REBLD_PRIM_FCB : L_NORM NOVALUE, ! rebuild primary fcb from header
: 347      1336 2      BUILD_EXT_FCBS : L_NORM NOVALUE, ! build extension fcb chain
: 348      1337 2      RELEASE_SERIAL_LOCK : L_NORM,      ! release file synchronization lock
: 349      1338 2      ALLOCATION_UNLOCK : L_NORM,      ! synchronize allocation/deallocation
: 350      1339 2      ARBITRATE_ACCESS : L JSB_2ARGS,      ! establish file access.
: 351      1340 2      CONV_ACCLOCK   : L_NORM,      ! convert/dequeue access lock.
: 352      1341 2      SERIAL_FILE    : L_NORM,      ! interlock file processing.
: 353      1342 2      GET_FIB        : L_NORM,      ! get FIB for operation
: 354      1343 2      GET_LOC_ATTR   : L_NORM,      ! get placement data form attribute list
: 355      1344 2      GET_LOC        : L_NORM,      ! get placement data
: 356      1345 2      SWITCH_VOLUME  : L_NORM,      ! switch context to specified volume
: 357      1346 2      SELECT_VOLUME  : L_NORM,      ! find volume in volume set for create
: 358      1347 2      CHECK_PROTECT  : L_NORM,      ! check file protection
: 359      1348 2      CHARGE_QUOTA   : L_NORM,      ! charge blocks to user's disk quota
: 360      1349 2      CREATE_HEADER  : L_NORM,      ! create a file ID and header
: 361      1350 2      CHECKSUM       : L_NORM,      ! compute header checksum
: 362      1351 2      MARK_DIRTY     : L_NORM,      ! mark buffer for write-back
: 363      1352 2      ACL_INIT_QUEUE : ADDRESSING_MODE (GENERAL), ! Initialize ACL queue
: 364      1353 2      ACL_ADDENTRY   : ADDRESSING_MODE (GENERAL), ! add entry to ACL
: 365      1354 2      ACL_BUILDACL   : ADDRESSING_MODE (GENERAL) L_NORM, ! build ACL into file headers
: 366      1355 2      READ_HEADER    : L_NORM,      ! read file header
: 367      1356 2      ENTER          : L_NORM,      ! enter file in directory
: 368      1357 2      COPY_NAME      : L_NORM,      ! copy file name to result string
: 369      1358 2      SET_REVISION    : L_NORM,      ! set file revision and exp dates
: 370      1359 2      CREATE_FCB     : L_NORM,      ! create an FCB
: 371      1360 2      CREATE_WINDOW  : L_NORM,      ! create a window
: 372      1361 2      SET_EXPIRE     : L_NORM,      ! enable expiration date recording
: 373      1362 2      MAKE_ACCESS    : L_NORM,      ! complete the access
: 374      1363 2      MARKDEL_FCB   : L_NORM,      ! mark FCB for delete
: 375      1364 2      WRITE_ATTRIB  : L_NORM,      ! write attributes
: 376      1365 2      EXTEND         : L_NORM,      ! extend the file
```



```

377 1366 2      SAVE CONTEXT      : L_NORM,      ! save reentrant context area
378 1367 2      RESTORE CONTEXT : L_NORM,      ! restore reentrant context area
379 1368 2      MARK_DELETE     : L_NORM,      ! mark file for delete
380 1369 2      REMAP_FILE      : L_NORM,      ! remap the file completely
381 1370 2      SEARCH_FCB      : L_NORM ADDRESSING_MODE (GENERAL); ! Search FCB list
382 1371 2
383 1372 2
384 1373 2      ! Enable the deaccess cleanup if an access is taking place.
385 1374 2      !
386 1375 2
387 1376 2      PACKET = .IO PACKET;
388 1377 2      FUNCTION = .PACKET[IRPSW_FUNC];
389 1378 2      IF .FUNCTION[IOSV_ACCESS]
390 1379 2      THEN
391 1380 2          BEGIN
392 1381 2              CLEANUP_FLAGS[CLF_ZCHANNEL] = 1;
393 1382 2              CLEANUP_FLAGS[CLF_DELWINDOW] = 1;
394 1383 2          END;
395 1384 2
396 1385 2      ! Set up pointers to interesting control blocks.
397 1386 2      !
398 1387 2
399 1388 2      PCB = .SCH$GL_PCBVEC[(IO PACKET[IRPSL_PID])<0,16>];
400 1389 2      ABD = .BBLOCK[.PACKET[IRPSL_SVAPTE], AIB$DESCRPT];
401 1390 2      FIB = GET_FIB (.ABD);      ! pointer to buffer descriptors
402 1391 2      ! pointer to FIB
403 1392 2
404 1393 2      IF .FIB[FIB$V_TRUNC]
405 1394 2      OR .FIB[FIB$V_VERLIMIT] GTRU 32767
406 1395 2      OR (.FUNCTION[IOSV_DELETE] AND NOT .FUNCTION[IOSV_ACCESS])
407 1396 2      OR (NOT .FUNCTION[IOSV_CREATE]
408 1397 2          AND (.FIB[FIB$V_EXTEND]
409 1398 2              OR .PACKET[IRPSW_BCNT] GTR ABD$C_ATTRIB
410 1399 2              OR .FUNCTION[IOSV_ACCESS]
411 1400 2          )
412 1401 2      )
413 1402 2      THEN ERR_EXIT (SS$BADPARAM);
414 1403 2
415 1404 2      IF .CURRENT_VCB[VCB$V_NOALLOC]
416 1405 2      THEN ERR_EXIT (SS$WRITLCK);
417 1406 2
418 1407 2      ! Do the create if requested. Start by allocating a file number from the
419 1408 2      ! index file bitmap and reading in the initial file header.
420 1409 2      !
421 1410 2
422 1411 2      IF .FUNCTION[IOSV_CREATE]
423 1412 2      THEN
424 1413 2          BEGIN
425 1414 2
426 1415 2      ! Deal with special cases related to create-if. Release any serialization
427 1416 2      ! lock we are holding, and force supersede mode to dispose of bad
428 1417 2      ! directory entries.
429 1418 2      !
430 1419 2
431 1420 2      IF .PACKET[IRPSV_FCODE] EQL IOSV_ACCESS
432 1421 2      THEN
433 1422 2          BEGIN
```



```

434 1423 4      IF .PRIM_LCKINDX NEQ 0
435 1424 4      THEN
436 1425 5          BEGIN
437 1426 5              RELEASE SERIAL_LOCK (.PRIM_LCKINDX);
438 1427 5              PRIM_LCKINDX = 0;
439 1428 4          END;
440 1429 4      FIB[FIB$V_SUPERSEDE] = 1;
441 1430 4
442 1431 4      ! Finally, the protection check if the directory has been accessed. This
443 1432 4      ! is because the protection check is not done in DIR_ACCESS (via ENTER) if
444 1433 4      ! the directory file has already been accessed.
445 1434 4      !
446 1435 4
447 1436 4      IF .DIR_FCB NEQ 0
448 1437 4      AND .CLEANUP_FLAGS[CLF_DIRECTORY]
449 1438 4      AND NOT .CLEANUP_FLAGS[CLF_SPOOLFILE]
450 1439 4      THEN
451 1440 5          BEGIN
452 1441 5              STATUS = CHECK_PROTECT (WRITE_ACCESS, 0, .DIR_FCB, 0,
453 1442 6                  (IF .BBLOCK [FIB[FIB$V_ALT_ACCESS], ARMSV_DELETE]
454 1443 5                  THEN ARMSM_WRITE ELSE 0),
455 1444 5                  .FIB[FIB$V_ALT_REQ]);
456 1445 5              IF .STATUS EQL SSS_NOTALLPRIV
457 1446 5              THEN FIB[FIB$V_ALT_GRANTED] = 0;
458 1447 4          END;
459 1448 4      END;
460 1449 4
461 1450 3      ! Handle any placement specified and find a suitable volume for the
462 1451 3      ! file in a volume set.
463 1452 3      !
464 1453 3
465 1454 3      FIB[FIB$V_PROPAGATE] = 0;
466 1455 3      IF .FIB[FIB$V_ALLOCATR]
467 1456 3      THEN GET LOC_ATTR (.ABD, .FIB);
468 1457 3      GET_LOC T.FIB, LOC_RVN, LOC_LBN;
469 1458 3      IF .LOC_RVN NEQ 0
470 1459 3      AND .FIB[FIB$V_EXACT]
471 1460 3      THEN
472 1461 3          SWITCH_VOLUME (.LOC_RVN)
473 1462 3      ELSE
474 1463 4          SELECT_VOLUME (.FIB, (IF .FIB[FIB$V_EXTEND]
475 1464 4              THEN .FIB[FIB$V_EXSZ]
476 1465 3              ELSE 0));
477 1466 3
478 1467 3      CHECK_PROTECT (CREATE_ACCESS, 0, 0, 0);
479 1468 3      IF .BBLOCK [CURRENT_UCB[UCB$V_DEVCHAR], DEV$V_SWL]
480 1469 3      OR .CURRENT_VCB[VCB$V_NOALLOC]
481 1470 3      THEN ERR_EXIT (SSS_WRTLCK);
482 1471 3
483 1472 3      HEADER = CREATE_HEADER (FIB[FIB$V_FID]);
484 1473 3
485 1474 3      ! Now build an initialized file header in the buffer.
486 1475 3      !
487 1476 3
488 1477 3      ARB = .PACKET[IRP$L_ARB];
489 1478 3
490 1479 3      IF .EXE$GL_DYNAMIC_FLAGS < EXE$V_CLASS_PROT, 1>
```



```
491 1480 3 THEN HEADER[FH2$B_IDOFFSET] = FH2$C_FULL_LENGTH / 2
492 1481 3 ELSE HEADER[FH2$B_IDOFFSET] = FH2$C_LENGTH / 2;
493 1482 3 HEADER[FH2$B_MPOFFSET] = .HEADER[FH2$B_IDOFFSET] + F12$C_LENGTH / 2;
494 1483 3 HEADER[FH2$B_ACOFFSET] = ($BYTEOFFSET (FH2$W_CHECKSUM)) / 2;
495 1484 3 HEADER[FH2$B_RSOFFSET] = ($BYTEOFFSET (FH2$W_CHECKSUM)) / 2;
496 1485 3 HEADER[FH2$W_SEG_NUM] = 0;
497 1486 3 HEADER[FH2$W_STROCLEV] = FH2$C_LEVEL2 + 1;
498 1487 3
499 1488 3 CH$FILL (0, 512 - $BYTEOFFSET(FH2$W_EXT_FID), HEADER[FH2$W_EXT_FID]);
500 1489 3 HEADER[FH2$L_FILEOWNER] = .ARB[ARB$L_UID];
501 1490 3 HEADER[FH2$W_FILEPROT] = .PCB[PCB$L_DEFPROT];
502 1491 3
503 1492 3 IF .FUNCTION[IOSV_DELETE]
504 1493 3 THEN HEADER[FH2$V_MARKDEL] = 1;
505 1494 3
506 1495 3 IF .CLEANUP_FLAGS[CLF_SPOOLFILE]
507 1496 3 THEN HEADER[FH2$V_SPOOL] = 1;
508 1497 3
509 1498 3 $ASSUME (ARB$S_CLASS EQL FH2$S_CLASS_PROT);
510 1499 3
511 1500 3 IF .EXESGL_DYNAMIC_FLAGS<EXESV_CLASS_PROT,1>
512 1501 3 THEN CH$MOVE (ARB$S_CLASS, ARB[ARB$R_CLASS], HEADER[FH2$R_CLASS_PROT]);
513 1502 3
514 1503 3 NEW_FID = 0; ! new file ID is no longer unrecorded
515 1504 3 CLEANUP_FLAGS[CLF_DELFIL] = 1;
516 1505 3 CLEANUP_FLAGS[CLF_HDRNOTCHG] = 1;
517 1506 3 FILE_HEADER = .HEADER; ! record header address for cleanup
518 1507 3 CHECKSUM (.HEADER);
519 1508 3
520 1509 3 ! At this point build the necessary FCB, even if the file is not accessed.
521 1510 3 ! This is necessary to allow the ACL to be built.
522 1511 3
523 1512 3
524 1513 3 FCB = KERNEL_CALL (CREATE_FCB, .HEADER);
525 1514 3 PRIMARY_FCB = .FCB;
526 1515 3 END;
527 1516 3
528 1517 3 ! If a non-zero directory ID is supplied, enter the file in the directory.
529 1518 3 ! Otherwise, just copy down the name string (if any) into the result string.
530 1519 3 ! Note that directory operations are also nooped on spool files operations.
531 1520 3
532 1521 3
533 1522 3 IF .CLEANUP_FLAGS[CLF_DIRECTORY] AND NOT .CLEANUP_FLAGS[CLF_SPOOLFILE]
534 1523 3 THEN
535 1524 3 BEGIN
536 1525 3 CH$FILL (0, FID$C_LENGTH, OLD_VERSION_FID);
537 1526 3 ENTER (.ABD, .FIB, RESULT_LENGTH, RESULT);
538 1527 3
539 1528 3 ! Always attempt to release the allocation lock here. We will be holding
540 1529 3 ! it if the directory was extended. It might make more sense to release
541 1530 3 ! it in the directory extension, but the call is relatively cheap.
542 1531 3
543 1532 3
544 1533 3 ALLOCATION_UNLOCK ();
545 1534 3
546 1535 3 ! ENTER may have flushed the new buffer from the cache if either the
547 1536 3 ! directory file header(s) and quota file header(s) were accessed and
```



```
548 1537 3 ! there were multiple headers. Make sure FILE_HEADER is what we think
549 1538 3 ! it is.
550 1539 3
551 1540 3
552 1541 3 IF .FUNCTION [IOSV_CREATE]
553 1542 3 THEN
554 1543 3 FILE_HEADER = READ_HEADER (0, .FCB);
555 1544 3
556 1545 3 IF .FUNCTION[IOSV_CREATE] OR .FIB[FIB$V_PROPAGATE]
557 1546 3 THEN
558 1547 3 BEGIN
559 1548 3
560 1549 4 ! If the CREATE modifier was not specified, then this must be a directory
561 1550 4 ! entry operation. In which case it is necessary to actually access the
562 1551 4 ! file being entered, so that an FCB will exist for the propagation to
563 1552 4 ! occur.
564 1553 4
565 1554 4 IF NOT .FUNCTION[IOSV_CREATE]
566 1555 4 THEN
567 1556 5 BEGIN
568 1557 5
569 1558 5 ! Switch context to the volume of the specified RVN.
570 1559 5
571 1560 5
572 1561 5 SWITCH_VOLUME (.FIB[FIB$W_FID_RVN]);
573 1562 5
574 1563 5 ! Synchronize further processing on this file.
575 1564 5
576 1565 5
577 1566 5 PRIM_LCKINDX = SERIAL_FILE (FIB [FIB$W_FID]);
578 1567 5
579 1568 5 ! Find the FCB of the file, if one exists. then read the file
580 1569 5 ! header. If there is no FCB, create one.
581 1570 5
582 1571 5
583 1572 5 FCB = SEARCH_FCB (FIB[FIB$W_FID]);
584 1573 5 HEADER = READ_HEADER (FIB[FIB$W_FID], .FCB);
585 1574 5 FCB_CREATED = 0;
586 1575 5
587 1576 5 IF .FCB EQL 0
588 1577 5 THEN
589 1578 6 BEGIN
590 1579 6 FCB_CREATED = 1;
591 1580 6 FCB = KERNEL_CALL (CREATE_FCB, .HEADER);
592 1581 5 END;
593 1582 5 PRIMARY_FCB = .FCB; ! record FCB for external use
594 1583 5
595 1584 5 ! If the file is multi-header, read the extension headers and create
596 1585 5 ! extension FCB's as necessary. Finally read back the primary header.
597 1586 5
598 1587 5
599 1588 5 IF .FCB_CREATED
600 1589 5 THEN
601 1590 5 BUILD_EXT_FCBS (.HEADER)
602 1591 5 ELSE
603 1592 5 IF .FCB [FCB$V_STALE]
604 1593 5 THEN
```



```

: 605      1594 6      BEGIN
: 606      1595 6      REBLD_PRIM_FCB (.PRIMARY_FCB, .HEADER);
: 607      1596 6      BUILD_EXT_FCBS (.HEADER);
: 608      1597 5      END;
: 609      1598 5
: 610      1599 5      ! Wipe out any acs that may have existed, because they are going
: 611      1600 5      ! to be propagated.
: 612      1601 5
: 613      1602 5
: 614      1603 5      IF .BBLOCK [FCB [FCB$R_ORB], ORB$V_ACL_QUEUE]
: 615      1604 5      THEN
: 616      1605 5          ACL_DELETEACL (FCB [FCB$L_ACLFL], 0);
: 617      1606 5
: 618      1607 4      END;
: 619      1608 4
: 620      1609 4      ! Now propagate the file attributes to the file just entered.
: 621      1610 4
: 622      1611 4          STATUS = PROPAGATE_ATTR (.FIB);
: 623      1612 4          IF NOT .STATUS THEN ERR_EXIT (.STATUS);
: 624      1613 4          HEADER = .FILE_HEADER;
: 625      1614 4          HEADER[FH2$L_FILEOWNER] = .PRIMARY_FCB[FCB$L_FILEOWNER];
: 626      1615 4          HEADER[FH2$W_FILEPROT] = .PRIMARY_FCB[FCB$W_FILEPROT];
: 627      1616 4          CHECKSUM (.HEADER);
: 628      1617 4          MARK_DIRTY (.HEADER);
: 629      1618 4          END;
: 630      1619 3      END
: 631      1620 2      ELSE
: 632      1621 3          BEGIN
: 633      1622 3              KERNEL_CALL (COPY_NAME, .ABD);
: 634      1623 3              RESULT_LENGTH = MINU (.ABD[ABD$C_NAME, ABD$W_COUNT], FI2$S_FILENAME+FI2$S_FILENAMEEXT);
: 635      1624 3              CH$MOVE (.RESULT_LENGTH,
: 636      1625 3                  .ABD[ABD$C_NAME, ABD$W_TEXT] + ABD[ABD$C_NAME, ABD$W_TEXT] + 1, RESULT);
: 637      1626 3          END;
: 638      1627 2
: 639      1628 2      ! Read the file header, regardless of the operation. Do a protection check
: 640      1629 2      ! on the directory pointed to by the present back link. If it is not valid,
: 641      1630 2      ! or if write access is allowed, then overwrite the back link with the new
: 642      1631 2      ! directory ID. Copy the file string into the header ident area. Then write
: 643      1632 2      ! attributes as specified.
: 644      1633 2
: 645      1634 2
: 646      1635 2      IF .FIB[FIB$W_FID_NUM] NEQ 65535
: 647      1636 2      OR .FIB[FIB$W_FID_SEQ] NEQ 65535
: 648      1637 2      OR .FIB[FIB$B_FID_NMX] NEQ 255
: 649      1638 2      THEN
: 650      1639 3          BEGIN
: 651      1640 3              PRIMARY_VCB = .CURRENT_VCB;
: 652      1641 3              IF .PRIMARY_VCB[VCB$W_RVN] NEQ 0
: 653      1642 3              THEN
: 654      1643 4                  BEGIN
: 655      1644 4                      UCB = .VECTOR [CURRENT_RVT[RVT$L_UCBLST], 0];
: 656      1645 4                      IF .UCB EQL 0
: 657      1646 4                      THEN ERR_EXIT (SS$DEVNOTMOUNT);
: 658      1647 4                      PRIMARY_VCB = .UCB[UCB$L_VCB];
: 659      1648 4                      END;
: 660      1649 3
: 661      1650 3      IF .PRIM_LCKINDX EQL 0
```



```

662 1651 3 THEN
663 1652 3     PRIM_LCKINDX = SERIAL_FILE (FIB [FIB$W_FID]);
664 1653 3
665 1654 3     HEADER = READ_HEADER (FIB[FIB$W_FID], 0);
666 1655 3     IDENT_AREA = .HEADER + .HEADER[FH2$B_IDOFFSET]*2;
667 1656 3
668 1657 3     CH$MOVE (FID$C_LENGTH, HEADER[FH2$W_BACKLINK], PREV_LINK);
669 1658 3     IF .PREV_LINK[FID$W_NUM] EQL 0
670 1659 3     AND .PREV_LINK[FID$W_RVN] EQL 0
671 1660 3     THEN
672 1661 4         BEGIN
673 1662 4             IF NOT .CLEANUP_FLAGS[CLF_SPOOLFILE]
674 1663 4             THEN
675 1664 5                 BEGIN
676 1665 5                     CH$MOVE (FID$C_LENGTH, FIB[FIB$W_DID], HEADER[FH2$W_BACKLINK]);
677 1666 5                     DEFAULT_RVN (HEADER[FH2$W_BK_FIDRVN], .CURRENT_RVN);
678 1667 5                     CLEANUP_FLAGS[CLF_FIXLINK] = 1;
679 1668 4                     END;
680 1669 4
681 1670 4                     CH$MOVE (F12$S_FILENAME, IDENT_AREA[F12$T_FILENAME], PREV_INAME);
682 1671 4                     CH$MOVE (F12$S_FILENAMEEXT, IDENT_AREA[F12$T_FILENAMEEXT],
683 1672 4                         PREV_INAME[F12$S_FILENAME]);
684 1673 4                     CH$COPY (.RESULT_LENGTH, RESULT, ' ', F12$S_FILENAME, IDENT_AREA[F12$T_FILENAME]);
685 1674 4                     IF .HEADER[FH2$B_MPOFFSET] - .HEADER[FH2$B_IDOFFSET]
686 1675 4                         GEQU ($BYTEOFFSET (F12$T_FILENAMEEXT) + F12$S_FILENAMEEXT) / 2
687 1676 4                     THEN
688 1677 5                         BEGIN
689 1678 5                             K = MAX (.RESULT_LENGTH - F12$S_FILENAME, 0);
690 1679 5                             CH$COPY (.K, RESULT[F12$S_FILENAME], ' ',
691 1680 5                                 F12$S_FILENAMEEXT, IDENT_AREA[F12$T_FILENAMEEXT]);
692 1681 4                             END;
693 1682 4
694 1683 4 ! Update revision count and date and expiration date as appropriate.
695 1684 4 !
696 1685 4
697 1686 4     SET_REVISION (.HEADER, 3);
698 1687 3     END;
699 1688 3
700 1689 3 ! Set up file dates; then write the attributes.
701 1690 3 !
702 1691 3
703 1692 3     IF .FUNCTION[IOSV_CREATE]
704 1693 3     THEN
705 1694 4         BEGIN
706 1695 4             IDENT_AREA[F12$W_REVISION] = 0;
707 1696 4             CH$MOVE (F12$S_CREDATE, IDENT_AREA[F12$Q_REVDATE], IDENT_AREA[F12$Q_CREDATE]);
708 1697 4
709 1698 4             IF .PACKET[IRP$W_BCNT] GTR ABD$C_ATTRIB
710 1699 4             THEN
711 1700 5                 BEGIN
712 1701 5                     WRITE_ATTRIB (.HEADER, .ABD, 0);
713 1702 5                     HEADER = .FILE_HEADER;
714 1703 4                     END;
715 1704 4
716 1705 4 ! If the file is now owned by a UIC other than the creator, add an ACL
717 1706 4 ! entry granting owner's access to the creator. Then write the modified
718 1707 4 ! ACL into the header.
```



```

: 719      1708 4
: 720      1709 4
: 721      1710 4
: 722      1711 4
: 723      1712 5
: 724      1713 5
: 725      1714 5
: 726      1715 5
: 727      1716 5
: 728      1717 5
: 729      1718 5
: 730      1719 5
: 731      1720 5
: 732      1721 5
: 733      1722 5
: 734      1723 5
: 735      1724 4
: 736      1725 4
: 737      1726 4
: 738      1727 4
: 739      1728 4
: 740      1729 4
: 741      1730 4
: 742      1731 4
: 743      1732 4
: 744      1733 4
: 745      1734 5
: 746      1735 5
: 747      1736 5
: 748      1737 5
: 749      1738 5
: 750      1739 5
: 751      1740 5
: 752      1741 5
: 753      1742 5
: 754      1743 5
: 755      1744 5
: 756      1745 6
: 757      1746 6
: 758      1747 6
: 759      1748 6
: 760      1749 6
: 761      1750 6
: 762      1751 6
: 763      1752 6
: 764      1753 5
: 765      1754 5
: 766      1755 5
: 767      1756 5
: 768      1757 5
: 769      1758 5
: 770      1759 5
: 771      1760 5
: 772      1761 4
: 773      1762 4
: 774      1763 4
: 775      1764 4

IF .HEADER[FH2$FILEOWNER] NEQ .ARB[ARB$UIC]
AND NOT .CLEANUP_FLAGS[CLF_SYSPRV]
THEN
    BEGIN
        ACL_INIT_QUEUE (PRIMARY_FCB[FCB$R_ORB]);
        ACL_CONTEXT = 0;
        ACE[ACESB_SIZE] = ACE_LENGTH;
        ACE[ACESB_TYPE] = ACESC_KEYID;
        ACE[ACESW_FLAGS] = ACESM_NOPROPAGATE;
        ACE[ACESL_ACCESS] = ACESM_CONTROL OR
            (.HEADER[FH2$W_FILEPROT])<4,4> XOR %B'1111';
        ACE[ACESL_KEY] = .ARB[ARB$UIC];
        ACL_ADDENTRY (PRIMARY_FCB[FCB$ACLFL], ACL_CONTEXT, ACE_LENGTH, ACE);
        STATUS = ACL_BUILDACL(.PRIMARY_FCB);
        IF NOT .STATUS THEN ERR_EXIT (.STATUS);
    END;

    CHARGE_QUOTA (.HEADER[FH2$FILEOWNER], 1, BITLIST (QUOTA_CHECK, QUOTA_CHARGE));
    CLEANUP_FLAGS[CLF_HDRNOTCHG] = 0;

    ! If access is requested, access the file.
    !
    IF .FUNCTION[IOSV_ACCESS]
    THEN
        BEGIN
            IF NOT ARBITRATE_ACCESS (.FIB [FIB$ACCTL], .FCB)
            THEN
                BUG_CHECK (XQPERR, 'how can we fail to access a new file?');

            CURRENT_WINDOW = CREATE_WINDOW (.FIB[FIB$ACCTL],
                .FIB[FIB$B_WSIZE], .HEADER, .PACKET[IRP$PID], .FCB);

            IF .CURRENT_WINDOW EQL 0
            THEN
                BEGIN
                    ! This will dequeue the access lock we may have taken above (if a cluster
                    ! device) because the refcnt will be zero.
                    !
                    CONV_ACCLOCK (0, .FCB);
                    ERR_EXIT (SS$EXBYTLM);
                    END;

                    MAKE_ACCESS (.FCB, .CURRENT_WINDOW, .ABD);

                    IF .FUNCTION[IOSV_DELETE]
                    THEN KERNEL_CALL (MARKDEL_FCB, .FCB);
                    IF .(PRIMARY_VCB[VCB$Q_RETAINMAX]+4) NEQ 0
                    THEN KERNEL_CALL (SET_EXPIRE);
                END;

                ! Now extend the file if requested.
                !

```



```

: 776      1765  4
: 777      1766  4      IF .FIB[FIB$V_EXTEND] THEN EXTEND (.FIB, .HEADER);
: 778      1767  4      HEADER = .FILE HEADER;
: 779      1768  4      KERNEL_CALL (UPDATE_FCB, .HEADER);
: 780      1769  3      END;
: 781      1770  3
: 782      1771  3      CHECKSUM (.HEADER);
: 783      1772  3      MARK_DIRTY (.HEADER);
: 784      1773  3
: 785      1774  4      IF (.FUNCTION[IO$V_CREATE] OR .FIB[FIB$V_PROPAGATE])
: 786      1775  3          AND .PRIMARY_FCB NEQ 0
: 787      1776  3      THEN
: 788      1777  3          IF .BBLOCK[PRIMARY_FCB[FCB$R_ORB], ORB$V_ACL_QUEUE]
: 789      1778  3              THEN
: 790      1779  4              BEGIN
: 791      1780  4                  STATUS = ACL_BUILDACL (.PRIMARY_FCB);
: 792      1781  4                  IF NOT .STATUS THEN ERR_EXIT (.STATUS);
: 793      1782  3              END;
: 794      1783  3
: 795      1784  3      ! Perform the remap operation if necessary to account for any initial extend.
: 796      1785  3      !
: 797      1786  3
: 798      1787  3      IF .FUNCTION[IO$V_ACCESS] AND .FIB[FIB$V_EXTEND]
: 799      1788  3      THEN IF .CURRENT_WINDOW[WCB$V_CATHEDRAL]
: 800      1789  3      THEN REMAP_FILE ();
: 801      1790  2      END;
: 802      1791  2
: 803      1792  2      ! If this is a supersede operation, delete the file that was removed during
: 804      1793  2      ! the enter operation above. This must be done last since we cannot undo
: 805      1794  2      ! a delete in cleaning up from a subsequent error. We first copy the primary
: 806      1795  2      ! context into the context save area since this is a secondary operation.
: 807      1796  2      !
: 808      1797  2
: 809      1798  2      IF .CLEANUP_FLAGS[CLF_SUPERSEDE]
: 810      1799  2      THEN
: 811      1800  3          BEGIN
: 812      1801  3              ALLOCATION_UNLOCK ();
: 813      1802  3              SAVE_CONTEXT ();
: 814      1803  3              CH$COPY (FID$C_LENGTH, SUPER_FID, 0,
: 815      1804  3                  FIB$C_LENGTH - $BYTEOFFSET (FIB$W_FID), SECOND_FIB[FIB$W_FID]);
: 816      1805  3              SECOND_FIB[FIB$B_AGENT_MODE] = .FIB[FIB$B_AGENT_MODE];
: 817      1806  3              MARK_DELETE (SECOND_FIB, 1, 0, 0);
: 818      1807  3              RESTORE_CONTEXT ();
: 819      1808  3          END;
: 820      1809  2
: 821      1810  2      RETURN 1;
: 822      1811  2
: 823      1812  2
: 824      1813  1      END;
```

! end of routine CREATE

```

.TITLE  CREATE
.IDENT  \V04-001\

.EXTRN  ACP$GB_WRITBACK
.EXTRN  SCH$GL_PCBVEC, EXE$GL_DYNAMIC_FLAGS
.EXTRN  EXE$V_CLASS_PROT
```



				.EXTRN	ACL_DELETEACL, UPDATE_FCB	
				.EXTRN	REBCD PRIM_FCB, BUILD_EXT_FCBS	
				.EXTRN	RELEASE_SERIAL_LOCK	
				.EXTRN	ALLOCATION_UNLOCK	
				.EXTRN	ARBITRATE_ACCESS	
				.EXTRN	CONV_ACCLOCK, SERIAL_FILE	
				.EXTRN	GET_FIB, GET_LOC_ATTR	
				.EXTRN	GET_LOC, SWITCH_VOLUME	
				.EXTRN	SELECT_VOLUME, CHECK_PROTECT	
				.EXTRN	CHARGE_QUOTA, CREATE_HEADER	
				.EXTRN	CHECKSUM, MARK_DIRTY	
				.EXTRN	ACL_INIT_QUEUE, ACL_ADDENTRY	
				.EXTRN	ACL_BUILDACL, READ_HEADER	
				.EXTRN	ENTER, COPY_NAME	
				.EXTRN	SET_REVISION, CREATE_FCB	
				.EXTRN	CREATE_WINDOW, SET_EXPIRE	
				.EXTRN	MAKE_ACCESS, MARKDEL_FCB	
				.EXTRN	WRITE_ATTRIB, EXTEND	
				.EXTRN	SAVE_CONTEXT, RESTORE_CONTEXT	
				.EXTRN	MARK_DELETE, REMAP_FILE	
				.EXTRN	SEARCH_FCB, BUGS_XOPERR	
				.PSECT	\$CODE\$,NOWRT,2	
				.ENTRY	CREATE, Save R2,R3,R4,R5,R6,R7,R8,R9,R11	: 1252
				MOVAB	-128(SP), SP	
				PUSHAB	4(BASE)	: 1328
				PUSHAB	8(BASE)	
				MOVAB	24(BASE), R9	
				PUSHAB	48(BASE)	
				PUSHAB	424(BASE)	
				PUSHAB	580(BASE)	
				PUSHL	-112(BASE)	: 1376
				ADDL3	#32, PACKET, R0	: 1377
				MOVZWL	(R0), FUNCTION	
				BBC	#6, FUNCTION, 1\$	: 1378
				BISW2	#1026, 2(BASE)	: 1382
				MOVL	@NSCH\$GL PCBVEC, R1	: 1388
				MOVL	-112(BASE), R0	
				ADDL2	#12, R0	
				MOVZWL	(R0), R0	
				MOVL	(R1)[R0], PCB	
				ADDL3	#44, PACKET, R0	: 1389
				MOVL	@(R0)+, ABD	
				PUSHL	ABD	: 1391
				CALLS	#1, GET_FIB	
				MOVL	R0, FIB	
				BLBS	23(FIB), 3\$	: 1393
				CMPL	44(FIB), #32767	: 1394
				BGTRU	3\$	
				BLBC	FUNCTION+1, 2\$	: 1395
				BBC	#6, FUNCTION, 3\$	
				TSTB	FUNCTION	: 1396
				BLSS	4\$	
				TSTB	22(FIB)	: 1397
				BLSS	3\$	
				ADDL3	#50, PACKET, R0	: 1398

  

				OBFC	00000	
5E	80	AE	9E	00002		
	04	AA	9F	00006		
	08	AA	9F	00009		
59	18	AA	9E	0000C		
	30	AA	9F	00010		
	01A8	CA	9F	00013		
	0244	CA	9F	00017		
	90	AA	DD	0001B		
50	6E	20	C1	0001E		
	7E	60	3C	00022		
06	6E	06	E1	00025		
	02	AA	8F	AB 00029		
	51	00000000G	9F	DO 0002F	1\$:	
	50	90	AA	DO 00036		
	50		OC	CO 0003A		
	50		60	3C 0003D		
	5B		6140	DO 00040		
50	04	AE	2C	C1 00044		
	56		90	DO 00049		
			56	DD 0004C		
	0000G	CF	01	FB 0004E		
	57		50	DO 00053		
	27	17	A7	EB 00056		
	7FFF	8F	A7	B1 0005A		
			1F	1A 00060		
	04	01	AE	E9 00062		
17	6E		06	E1 00066		
			6E	95 0006A	2\$:	
			16	19 0006C		
			A7	95 0006E		
			0E	19 00071		
50	04	AE	32	C1 00073		



		05	60	B1	00078	CMPW	(R0), #5		
			04	1A	0007B	BGTRU	3\$		
	03	6E	06	E1	0007D	BBC	#6, FUNCTION, 4\$		1399
			14	BF	00081	CHMU	#20		1402
				04	00083	RET			
		50	98	AA	D0	00084	4\$: MOVL	-104(BASE), R0	1404
	03	0B	A0	04	E1	00088	BBC	#4, 11(R0), 5\$	
			00C2	31	0008D	BRW	16\$		
			6E	95	00090	5\$: TSTB	FUNCTION		1411
			03	19	00092	BLSS	6\$		
			015E	31	00094	BRW	23\$		
	50	04	AE	20	C1	00097	6\$: ADDL3	#32, PACKET, R0	1420
32	60	06		00	ED	0009C	CMPZV	#0, #6, (R0), #50	
				4F	12	000A1	BNEQ	10\$	
				69	D5	000A3	TSTL	(R9)	1423
				09	13	000A5	BEQL	7\$	
				69	DD	000A7	PUSHL	(R9)	1426
	0000G	CF		01	FB	000A9	CALLS	#1, RELEASE_SERIAL_LOCK	
				69	D4	000AE	CLRL	(R9)	1427
	15	A7		04	88	000B0	7\$: BISB2	#4, 21(FIB)	1429
		50	00D0	CA	D0	000B4	MOVL	208(BASE), R0	1436
				37	13	000B9	BEQL	10\$	
	33	6A		06	E1	000BB	BBC	#6, (BASE), 10\$	1437
				6A	95	000BF	TSTB	(BASE)	1438
				2F	19	000C1	BLSS	10\$	
7E	38	A7		00	EF	000C3	EXTZV	#0, #1, 56(FIB), -(SP)	1444
		04	3C	03	E1	000C9	BBC	#3, 60(FIB), 8\$	1442
				02	DD	000CE	PUSHL	#2	
				02	11	000D0	BRB	9\$	
				7E	D4	000D2	8\$: CLRL	-(SP)	
				7E	D4	000D4	9\$: CLRL	-(SP)	1441
				50	DD	000D6	PUSHL	R0	
				01	7D	000D8	MOVQ	#1, -(SP)	
	0000G	7E		06	FB	000DB	CALLS	#6, CHECK_PROTECT	
	24	CF		50	D0	000E0	MOVL	R0, STATUS	
	00000681	AE	24	AE	D1	000E4	CMPL	STATUS, #1665	1445
		8F		04	12	000EC	BNEQ	10\$	
				02	8A	000EE	BICB2	#2, 56(FIB)	1446
	38	A7		08	8A	000F2	10\$: BICB2	#8, 56(FIB)	1454
	38	A7		04	E1	000F6	BBC	#4, 22(FIB), 11\$	1455
08	16	A7		56	7D	000FB	MOVQ	ABD, -(SP)	1456
		7E		02	FB	000FE	CALLS	#2, GET_LOC_ATTR	
	0000G	CF		20	AA	9F	11\$: PUSHAB	32(BASE)	1457
				1C	AA	9F	PUSHAB	28(BASE)	
				57	DD	00109	PUSHL	FIB	
	0000G	CF		03	FB	0010B	CALLS	#3, GET_LOC	
				1C	AA	D5	TSTL	28(BASE)	1458
				0E	13	00113	BEQL	12\$	
		0A		20	A7	E9	12\$: BLBC	32(FIB), 12\$	1459
				1C	AA	DD	PUSHL	28(BASE)	1461
	0000G	CF		01	FB	0011C	CALLS	#1, SWITCH_VOLUME	
				13	11	00121	BRB	15\$	
				16	A7	95	12\$: TSTB	22(FIB)	1463
				05	18	00126	BGEQ	13\$	
				18	A7	DD	PUSHL	24(FIB)	1464
				02	11	0012B	BRB	14\$	
				7E	D4	0012D	13\$: CLRL	-(SP)	1463



		0000G	CF			57	DD	0012F	14\$:	PUSHL	FIB		
						02	FB	00131		CALLS	#2, SELECT_VOLUME		
			7E			7E	7C	00136	15\$:	CLRQ	-(SP)		1467
		0000G	CF			03	7D	00138		MOVQ	#3, -(SP)		
			50	94		04	FB	00138		CALLS	#4, CHECK_PROTECT		
09		3B	A0			AA	D0	00140		MOVL	-108(BASE), R0		1468
			50	98		01	E0	00144		BBS	#1, 59(R0), 16\$		
05		0B	A0			AA	D0	00149		MOVL	-104(BASE), R0		1469
						04	E1	0014D		BBC	#4, 11(R0), 17\$		
				025C		8F	BF	00152	16\$:	CHMU	#604		1470
							04	00156		RET			
				04		A7	9F	00157	17\$:	PUSHAB	4(FIB)		1472
		0000G	CF			01	FB	0015A		CALLS	#1, CREATE_HEADER		
			58			50	D0	0015F		MOVL	R0, HEADER		
50		04	AE	00000058		8F	C1	00162		ADDL3	#88, PACKET, R0		1477
		1C	AE			60	D0	0016B		MOVL	(R0), ARB		
05		00000000G	9F	00000000G		8F	E1	0016F		BBC	#EXESV_CLASS_PROT, @#EXESGL_DYNAMIC_FLAGS, -		1479
											18\$		
			68			36	90	0017B		MOVB	#54, (HEADER)		1480
						03	11	0017E		BRB	19\$		
01	A8		68			28	90	00180	18\$:	MOVB	#40, (HEADER)		1481
		02	A8	FFFF		3C	81	00183	19\$:	ADDB3	#60, (HEADER), 1(HEADER)		1482
		06	A8	0201		8F	3C	00188		MOVZWL	#65535, 2(HEADER)		1483
01F2	8F		6E			8F	B0	0018E		MOVW	#513, 6(HEADER)		1486
				0E		00	2C	00194		MOVCS	#0, (SP), #0, #498, 14(HEADER)		1488
		50	1C	AE		A8		0019B					
			3C	A8		38	C1	0019D		ADDL3	#55, ARB, R0		1489
			40	A8	0114	60	D0	001A2		MOVL	(R0), 60(HEADER)		
			05	A8	01	CB	B0	001A6		MOVW	276(PCB), 64(HEADER)		1490
		35	A8	80		AE	E9	001AC		BLBC	FUNCTION+1, 20\$		1492
						8F	88	001B0		BISB2	#128, 53(HEADER)		1493
						6A	95	001B5	20\$:	TSTB	(BASE)		1495
						04	18	001B7		BGEQ	21\$		
		35	A8			10	88	001B9		BISB2	#16, 53(HEADER)		1496
0A		00000000G	9F	00000000G		8F	E1	001BD	21\$:	BBC	#EXESV_CLASS_PROT, @#EXESGL_DYNAMIC_FLAGS, -		1500
											22\$		
58	5B	1C	AE			0C	C1	001C9		ADDL3	#12, ARB, R11		1501
	A8		6B			14	28	001CE		MOVCS	#20, (R11), 88(HEADER)		
				A8		AA	D4	001D3	22\$:	CLRL	-88(BASE)		1503
		02	AA	0820		8F	A8	001D6		BISW2	#2080, 2(BASE)		1505
		18	BE			58	DC	001DC		MOVL	HEADER, @24(SP)		1506
						58	DD	001E0		PUSHL	HEADER		1507
		0000G	CF			01	FB	001E2		CALLS	#1, CHECKSUM		
						58	DD	001E7		PUSHL	HEADER		1513
		0000G	CF			01	FB	001E9		CALLS	#1, CREATE_FCB		
			5B			50	D0	001EE		MOVL	R0, FCB		
		14	BE			5B	D0	001F1		MOVL	FCB, @20(SP)		1514
03			6A			06	E0	001F5	23\$:	BBS	#6, (BASE), 25\$		1522
						00E7	31	001F9	24\$:	BRW	33\$		
						6A	95	001FC	25\$:	TSTB	(BASE)		
						F9	19	001FE		BLSS	24\$		
06		00	6E			00	2C	00200		MOVCS	#0, (SP), #0, #6, 332(BASE)		1525
				014C		CA		00205					
				44		AE	9F	00208		PUSHAB	RESULT		1526
				2C		AE	9F	0020B		PUSHAB	RESULT_LENGTH		
			7E			56	7D	0020E		MOVQ	ABD, -(SP)		
		0000G	CF			04	FB	00211		CALLS	#4, ENTER		



0000G	CF	00	FB	00216	CALLS	#0, ALLOCATION_UNLOCK	1533
		6E	95	0021B	TSTB	FUNCTION	1541
		0D	18	0021D	BGEQ	26\$	
		5B	DD	0021F	PUSHL	FCB	1543
		7E	D4	00221	CLRL	-(SP)	
0000G	CF	02	FB	00223	CALLS	#2, READ_HEADER	
18	BE	50	D0	00228	MOVL	R0, @24(SP)	
		6E	95	0022C	TSTB	FUNCTION	1545
		08	19	0022E	BLSS	27\$	
03	38	A7	03	E0	BBS	#3, 56(FIB), 27\$	
			00D4	31	BRW	35\$	
			6E	95	TSTB	FUNCTION	1554
			6F	19	BLSS	31\$	
	7E	08	A7	3C	MOVZWL	8(FIB), -(SP)	1561
0000G	CF		01	FB	CALLS	#1, SWITCH_VOLUME	
		04	A7	9F	PUSHAB	4(FIB)	1566
0000G	CF		01	FB	CALLS	#1, SERIAL_FILE	
69			50	D0	MOVL	R0, (R9)	
		04	A7	9F	PUSHAB	4(FIB)	1572
00000000G	00		01	FB	CALLS	#1, SEARCH_FCB	
5B			50	D0	MOVL	R0, FCB	
		04	5B	DD	PUSHL	FCB	1573
			A7	9F	PUSHAB	4(FIB)	
0000G	CF		02	FB	CALLS	#2, READ_HEADER	
5B			50	D0	MOVL	R0, HEADER	
			52	D4	CLRL	FCB_CREATED	1574
			5B	D5	TSTL	FCB	1576
			0D	12	BNEQ	28\$	
	52		01	D0	MOVL	#1, FCB_CREATED	1579
			58	DD	PUSHL	HEADER	1580
0000G	CF		01	FB	CALLS	#1, CREATE_FCB	
5B			50	D0	MOVL	R0, FCB	
14	BE		5B	D0	MOVL	FCB, @20(SP)	1582
	OE		52	E8	BLBS	FCB_CREATED, 29\$	1588
	11	23	AB	E9	BLBC	35(FCB), 30\$	1592
			58	DD	PUSHL	HEADER	1595
		18	BE	DD	PUSHL	@24(SP)	
0000G	CF		02	FB	CALLS	#2, REBLD_PRIM_FCB	
			58	DD	PUSHL	HEADER	1596
0000G	CF		C1	FB	CALLS	#1, BUILD_EXT_FCBS	
63	AB		01	E1	BBC	#1, 99(FCB), 31\$	1603
			7E	D4	CLRL	-(SP)	1605
		0080	CB	9F	PUSHAB	128(FCB)	
00000000G	00		02	FB	CALLS	#2, ACL_DELETEACL	
			57	DD	PUSHL	FIB	1611
0000V	CF		01	FB	CALLS	#1, PROPAGATE_ATTR	
24	AE		50	D0	MOVL	R0, STATUS	
	03	24	AE	E8	BLBS	STATUS, 32\$	1612
			027E	31	BRW	55\$	
	58	18	BE	D0	MOVL	@24(SP), HEADER	1613
	50	14	BE	D0	MOVL	@20(SP), R0	1614
3C	A8	58	A0	D0	MOVL	88(R0), 60(HEADER)	
	50	14	BE	D0	MOVL	@20(SP), R0	1615
40	A8	70	A0	B0	MOVW	112(R0), 64(HEADER)	
			58	DD	PUSHL	HEADER	1616
0000G	CF		01	FB	CALLS	#1, CHECKSUM	
			58	DD	PUSHL	HEADER	1617



		0000G	CF		01	FB	002DC	CALLS	#1, MARK_DIRTY		
					29	11	002E1	BRB	35\$		1522
		0000G	CF		56	DD	002E3	PUSHL	ABD		1622
			50		01	FB	002E5	CALLS	#1, COPY_NAME		
		0056	8F	12	A6	3C	002EA	MOVZWL	18(ABD), -R0		1623
					50	B1	002EE	CMPW	R0, #86		
			50		04	1B	002F3	BLEQU	34\$		
			28		56	8F	9A	002F5	MOVZBL	#86, R0	
			51		50	D0	002F9	MOVL	R0, RESULT_LENGTH		
			50		10	A6	9E	002FD	MOVAB	16(ABD), RT	1625
			50		61	3C	00301	MOVZWL	(R1), R0		
44	AE	01	A1		28	AE	28	00304	MOVCL	RESULT_LENGTH, 1(R1)[R0], RESULT	1624
		FFFF	8F		04	A7	B1	0030C	CMPW	4(FIB), #65535	1635
					12	12	00312	BNEQ	36\$		
		FFFF	8F		06	A7	B1	00314	CMPW	6(FIB), #65535	1636
					0A	12	0031A	BNEQ	36\$		
		FF	8F		09	A7	91	0031C	CMPB	9(FIB), #255	1637
					03	12	00321	BNEQ	36\$		
					0230	31	00323	BRW	57\$		
		20	AE		98	AA	D0	00326	MOVL	-104(BASE), PRIMARY_VCB	1640
	50	20	AE			0E	C1	0032B	ADDL3	#14, PRIMARY_VCB, R0	1641
					60	B5	00330	TSTW	(R0)		
			50		14	13	00332	BEQL	38\$		
			51		9C	AA	D0	00334	MOVL	-100(BASE), R0	1644
					44	A0	D0	00338	MOVL	68(R0), UCB	
					05	12	0033C	BNEQ	37\$		1645
					007C	8F	BF	0033E	CHMU	#124	1646
						04	00342	RET			
		20	AE		34	A1	D0	00343	MOVL	52(UCB), PRIMARY_VCB	1647
					69	D5	00348	TSTL	(R9)		1650
					0B	12	0034A	BNEQ	39\$		
					04	A7	9F	0034C	PUSHAB	4(FIB)	1652
		0000G	CF		01	FB	0034F	CALLS	#1, SERIAL_FILE		
			69		50	D0	00354	MOVL	R0, (R9)		
					7E	D4	00357	CLRL	-(SP)		1654
					04	A7	9F	00359	PUSHAB	4(FIB)	
		0000G	CF		02	FB	0035C	CALLS	#2, READ_HEADER		
			58		50	D0	00361	MOVL	R0, HEADER		
			50		68	9A	00364	MOVZBL	(HEADER), R0		1655
			59		6840	3E	00367	MOVAB	(HEADER)[R0], IDENT_AREA		
10	BE	42	A8		06	28	0036B	MOVCL	#6, 66(HEADER), @16(SP)		1657
					10	BE	B5	00371	TSTW	@16(SP)	1658
					68	12	00374	BNEQ	44\$		
	50	10	AE		04	C1	00376	ADDL3	#4, 16(SP), R0		1659
					60	B5	0037B	TSTW	(R0)		
					5F	12	0037D	BNEQ	44\$		
					6A	95	0037F	TSTB	(BASE)		1662
					17	19	00381	BLSS	41\$		
					06	28	00383	MOVCL	#6, 10(FIB), 66(HEADER)		1665
AO	AA	42	A8	0A	A7			CMPZV	#0, #8, 70(HEADER), -96(BASE)		1666
		46	A8	08	08			BNEQ	40\$		
					03	12	00390	CLRB	70(HEADER)		
					46	A8	94	00392	BISB2	#64, 3(BASE)	1667
					40	8F	88	00395	MOVCL	#20, (IDENT_AREA), @12(SP)	1670
					14	28	0039A	ADDL3	#20, 12(SP), -(SP)		1672
					14	C1	0039F	MOVCL	#66, 54(IDENT_AREA), @ (SP)+		
					0042	8F	28	003A4	MOVCL	RESULT_LENGTH, RESULT, #32, #20, -	1673
					28	AE	2C	003AB			



0042	8F	20	58	AE	36	50	2C	003CB	42\$:	MOVCS	K, RESULT+20, #32, #66, 54(IDENT_AREA)	1680
					03	DD	003D5	43\$:	PUSHL	#3		1686
					58	DD	003D7		PUSHL	HEADER		
					02	FB	003D9		CALLS	#2, SET REVISION		
					6E	95	003DE	44\$:	TSTB	FUNCTION		1692
					03	19	003E0		BLSS	45\$		
					01	1F	003E2	45\$:	BRW	53\$		
					14	A9	B4	003E5		CLRW	20(IDENT_AREA)	1695
						08	28	003E8		MOVC3	#8, 30(IDENT_AREA), 22(IDENT_AREA)	1696
						32	C1	003EE		ADDL3	#50, PACKET, RO	1698
						60	B1	003F3		CMPW	(RO), #5	
						0F	1B	003F6		BLEQU	46\$	
						7E	D4	003F8		CLRL	-(SP)	1701
						56	DD	003FA		PUSHL	ABD	
						58	DD	003FC		PUSHL	HEADER	
						03	FB	003FE		CALLS	#3, WRITE ATTRIB	
					18	BE	D0	00403		MOVL	@24(SP), HEADER	1702
						38	C1	00407	46\$:	ADDL3	#56, ARB, RO	1709
					3C	A8	D1	0040C		CMP	60(HEADER), (RO)	
						63	13	00410		BEQL	47\$	
					01	AA	E8	00412		BLBS	1(BASE), 47\$	1710
					58	8F	C1	00416		ADDL3	#88, @20(SP), -(SP)	1713
						01	FB	0041F		CALLS	#1, ACL_INIT_QUEUE	
					2C	AE	D4	00426		CLRL	ACL_CONTEXT	1714
						8F	D0	00429		MOVL	#134217996, ACE	1715
					10C	04	EF	00431		EXTZV	#4, #4, 64(HEADER), RO	1719
						0F	CC	00437		XORL2	#15, RO	
						10	C9	0043A		BISL3	#16, RO, ACE+4	1718
						38	C1	0043F		ADDL3	#55, ARB, RO	1720
						60	D0	00444		MOVL	(RO), ACE+8	
					30	AE	9F	00448		PUSHAB	ACE	1721
						0C	DD	0044B		PUSHL	#12	
					34	AE	9F	0044D		PUSHAB	ACL_CONTEXT	
					80	8F	C1	00450		ADDL3	#128, @32(SP), -(SP)	
						04	FB	00459		CALLS	#4, ACL_ADDENTRY	
					14	BE	DD	00460		PUSHL	@20(SP)	1722
						01	FB	00463		CALLS	#1, ACL_BUILDACL	
						50	D0	0046A		MOVL	RO, STATUS	
					24	AE	E8	0046E		BLBS	STATUS, 47\$	1723
						00	C6	31	00472	BRW	55\$	
						03	DD	00475	47\$:	PUSHL	#3	1726
						01	DD	00477		PUSHL	#1	
					3C	A8	DD	00479		PUSHL	60(HEADER)	
						03	FB	0047C		CALLS	#3, CHARGE_QUOTA	
						08	8A	00481		BICB2	#8, 3(BASE)	1727
						06	E1	00485		BBC	#6, FUNCTION, 51\$	1732
						5B	D0	00489		MOVL	FCB, R1	1736



50			67	DO	0048C	MOVL	(FIB), R0			
			0000G	30	0048F	BSBW	ARBITRATE_ACCESS			
04			50	E8	00492	BLBS	R0, 48\$			
				FEFF	00495	BUGW		1738		
				0000*	00497	.WORD	<BUG\$_XQPERR!4>			
52	08	AE	5B	DD	00499	48\$:	PUSHL	FCB	1741	
			0C	C1	0049B		ADDL3	#12, PACKET, R2		
			62	DD	004A0		PUSHL	(R2)		
			58	DD	004A2		PUSHL	HEADER		
	7E		A7	98	004A4		CVTBL	3(FIB), -(SP)		
			67	DD	004A8		PUSHL	(FIB)	1740	
	0000G	CF	05	FB	004AA		CALLS	#5, CREATE_WINDOW		
	0C	AA	50	DO	004AF		MOVL	R0, 12(BASE)		
			0E	12	004B3		BNEQ	49\$	1743	
			5B	DD	004B5		PUSHL	FCB	1751	
			7E	D4	004B7		CLRL	-(SP)		
	0000G	CF	02	FB	004B9		CALLS	#2, CONV_ACCLOCK		
			2A14	8F	004BE		CHMU	#10772	1752	
				04	004C2		RET			
			56	DD	004C3	49\$:	PUSHL	ABD	1755	
			0C	AA	004C5		PUSHL	12(BASE)		
			5B	DD	004C8		PUSHL	FCB		
	0000G	CF	03	FB	004CA		CALLS	#3, MAKE_ACCESS		
		07	01	AE	004CF		BLBC	FUNCTION#1, 50\$	1757	
			5B	DD	004D3		PUSHL	FCB	1758	
50	0000G	CF	01	FB	004D5		CALLS	#1, MARK_DEL_FCB		
	20	AE	8F	C1	004DA	50\$:	ADDL3	#120, PRIMARY_VCB, R0	1759	
			60	D5	004E3		TSTL	(R0)		
			05	13	004E5		BEQL	51\$		
	0000G	CF	00	FB	004E7		CALLS	#0, SET_EXPIRE	1760	
			16	A7	95	004EC	51\$:	TSTB	22(FIB)	1766
			08	18	004EF		BGEQ	52\$		
			57	7D	004F1		MOVQ	FIB, -(SP)		
	0000G	CF	02	FB	004F4		CALLS	#2, EXTEND		
		58	18	BE	004F9	52\$:	MOVL	@24(SP), HEADER	1767	
			58	DD	004FD		PUSHL	HEADER	1768	
	0000G	CF	01	FB	004FF		CALLS	#1, UPDATE_FCB		
			58	DD	00504	53\$:	PUSHL	HEADER	1771	
	0000G	CF	01	FB	00506		CALLS	#1, CHECKSUM		
			58	DD	00508		PUSHL	HEADER	1772	
	0000G	CF	01	FB	0050D		CALLS	#1, MARK_DIRTY		
			6E	95	00512		TSTB	FUNCTION	1774	
			05	19	00514		BLSS	54\$		
24	38	A7	03	E1	00516		BBC	#3, 56(FIB), 56\$		
			14	BE	0051B	54\$:	TSTL	@20(SP)	1775	
			1F	13	0051E		BEQL	56\$		
			14	BE	00520		MOVL	@20(SP), R0	1777	
16	63	50	01	E1	00524		BBC	#1, 99(R0), 56\$		
		A0	14	BE	00529		PUSHL	@20(SP)	1780	
	00000000G	00	01	FB	0052C		CALLS	#1, ACL_BUILDACL		
	24	AE	50	DO	00533		MOVL	R0, STATUS		
		04	24	AE	E8	00537	BLBS	STATUS, 56\$	1781	
			24	AE	BF	0053B	55\$:	CHMU	STATUS	
				04	0053E		RET			
13		6E	06	E1	0053F	56\$:	BBC	#6, FUNCTION, 57\$	1787	
			16	A7	95	00543	TSTB	22(FIB)		
				0E	18	00546	BGEQ	57\$		



CREATE  
V04-001

I 1  
16-Sep-1984 00:06:06  
14-Sep-1984 12:30:13

VAX-11 Bliss-32 V4.0-742  
DISK\$VMSMASTER:[F11X.SRC]CREATE.B32;2

Page 23  
(2)

05	0B	50	0C	AA	D0	00548	MOVL	12(BASE), R0	:	1788
	0000G	A0		06	E1	0054C	BBC	#6, 11(R0), 57\$	:	
31		CF		00	FB	00551	CALLS	#0, REMAP FILE	:	1789
	0000G	6A		05	E1	00556	BBC	#5, (BASE), 58\$	:	1798
	0000G	CF		00	FB	0055A	CALLS	#0, ALLOCATION UNLOCK	:	1801
56	08	AE		00	FB	0055F	CALLS	#0, SAVE_CONTEXT	:	1802
00	01FE	CA		04	C1	00564	ADDL3	#4, 8(SPT, R6	:	1804
				06	2C	00569	MOVCS	#6, 510(BASE), #0, #60, (R6)	:	
50	08	AE		66		00570			:	
		60	2E	2E	C1	00571	ADDL3	#46, 8(SP), R0	:	1805
				A7	90	00576	MOVB	46(FIB), (R0)	:	
				7E	7C	0057A	CLRQ	-(SP)	:	1806
			14	01	DD	0057C	PUSHL	#1	:	
	0000G	CF		AE	DD	0057E	PUSHL	20(SP)	:	
	0000G	CF		04	FB	00581	CALLS	#4, MARK_DELETE	:	
		50		00	FB	00586	CALLS	#0, RESTORE_CONTEXT	:	1807
				01	D0	0058B	MOVL	#1, R0	:	1811
				04	0058E		RET		:	1813

; Routine Size: 1423 bytes, Routine Base: \$CODE\$ + 0000



```

: 826 1814 1 ROUTINE PROPAGATE_ATTR (FIB) : L_NORM =
: 827 1815 1
: 828 1816 1 ++
: 829 1817 1
: 830 1818 1 FUNCTIONAL DESCRIPTION:
: 831 1819 1
: 832 1820 1 This routine is called to propagate the file attributes from one
: 833 1821 1 file to another. This may be from one version of a file to another
: 834 1822 1 version of the file (either higher or lower) or from the parent
: 835 1823 1 directory to the newly created file. The following attributes are
: 836 1824 1 currently copied:
: 837 1825 1 1) File owner UIC
: 838 1826 1 2) File Access Control List (ACL)
: 839 1827 1 3) File protection (With some twiddling)
: 840 1828 1
: 841 1829 1 CALLING SEQUENCE:
: 842 1830 1 PROPAGATE_ATTR (ARG1)
: 843 1831 1
: 844 1832 1 INPUT PARAMETERS:
: 845 1833 1 ARG1: address of the supplied FIB
: 846 1834 1
: 847 1835 1 IMPLICIT INPUTS:
: 848 1836 1 PRIMARY_FCB: address of the new file's FCB
: 849 1837 1 DIR_FCB: address of the directory file's FCB
: 850 1838 1 OLD_VERSION_FID: FID of the old version of the file
: 851 1839 1
: 852 1840 1 OUTPUT PARAMETERS:
: 853 1841 1 none
: 854 1842 1
: 855 1843 1 IMPLICIT OUTPUTS:
: 856 1844 1 none
: 857 1845 1
: 858 1846 1 ROUTINE VALUE:
: 859 1847 1 1 if success
: 860 1848 1 error code otherwise
: 861 1849 1
: 862 1850 1 SIDE EFFECTS:
: 863 1851 1 The attributes in the file header of the new file are modified
: 864 1852 1 according to the attribute of the old version or parent directory.
: 865 1853 1
: 866 1854 1 --
: 867 1855 1
: 868 1856 2 BEGIN
: 869 1857 2
: 870 1858 2 MAP
: 871 1859 2 FIB : REF BBLOCK; ! Address of the FIB
: 872 1860 2
: 873 1861 2 LOCAL
: 874 1862 2 STATUS, ! Routine exit status
: 875 1863 2 WINDOW : REF BBLOCK, ! Address of created window
: 876 1864 2 FILE_FCB : REF BBLOCK, ! FCB for newly created file
: 877 1865 2 FCB : REF BBLOCK; ! Address of FCB from window
: 878 1866 2
: 879 1867 2 BIND_COMMON;
: 880 1868 2
: 881 1869 2 EXTERNAL ROUTINE
: 882 1870 2 READ_HEADER : L_NORM, ! read file header
```



```

: 883      1871 2      SAVE_CONTEXT : L_NORM,      ! Save reentrant context area
: 884      1872 2      RESTORE_CONTEXT : L_NORM,    ! Restore reentrant context area
: 885      1873 2      OPEN_FILE : L_NORM,          ! Open a file
: 886      1874 2      CLOSE_FILE : L_NORM,         ! Close a file
: 887      1875 2      CHECK_PROTECT : L_NORM;       ! Perform a protection check
: 888      1876 2
: 889      1877 2      ENABLE PROPAGATE_HANDLER;
: 890      1878 2
: 891      1879 2
: 892      1880 2      ! What we do depends on whether there is an old version present.
: 893      1881 2      ! If it exists, we copy attributes from it. If not, we copy attributes
: 894      1882 2      ! from the directory. If the old version is the same as the file being
: 895      1883 2      ! entered, we do nothing, because the net effect would be a NOP anyway,
: 896      1884 2      ! and we can't open the same file in both primary and secondary context.
: 897      1885 2      !
: 898      1886 2
: 899      1887 2      IF CH$EQL (FID$C_LENGTH, OLD_VERSION_FID,
: 900      1888 2          FID$C_LENGTH, PRIMARY_FCB[FCB$W_FID])
: 901      1889 2      THEN RETURN 1;
: 902      1890 2
: 903      1891 2      IF .OLD_VERSION_FID[FID$W_NUM] NEQ 0
: 904      1892 2      OR .OLD_VERSION_FID[FID$B_NMX] NEQ 0
: 905      1893 2      THEN
: 906      1894 3          BEGIN
: 907      1895 3              LOCAL SAVCURRINDX;
: 908      1896 3              SAVE_STATUS = .USER_STATUS;
: 909      1897 3              FILE_FCB = .PRIMARY_FCB;
: 910      1898 3              SAVCURRINDX = .CURR_LCKINDX;
: 911      1899 3              SAVE_CONTEXT ();
: 912      1900 3              WINDOW = OPEN_FILE (OLD_VERSION_FID, 2);
: 913      1901 3              IF .WINDOW NEQ 0
: 914      1902 3              THEN
: 915      1903 4                  BEGIN
: 916      1904 4                      FCB = .WINDOW[WCB$L_FCB];
: 917      1905 4                      IF CHECK_PROTECT (RDATT_ACCESS, 0, .PRIMARY_FCB,
: 918      1906 4                          MAXU (.IO_PACKET[IRP$V_MODE], .FIB[FIB$B_AGENT_MODE]))
: 919      1907 4                      THEN
: 920      1908 5                          BEGIN
: 921      1909 5
: 922      1910 5          ! Restore the current lock index we had from primary context.
: 923      1911 5          ! COPY_INFO may need to read the primary file's headers.
: 924      1912 5          !
: 925      1913 5
: 926      1914 5              CURR_LCKINDX = .SAVCURRINDX;
: 927      1915 5              STATUS = KERNEL_CALL (COPY_INFO, .FCB, .FILE_FCB, .FIB, 0);
: 928      1916 5              CLOSE_FILE (.WINDOW);
: 929      1917 5              RESTORE_CONTEXT ();
: 930      1918 5              READ_HEADER (CURRENT_FIB[FIB$W_FID], .PRIMARY_FCB);
: 931      1919 5              RETURN .STATUS;
: 932      1920 4              END;
: 933      1921 3          END;
: 934      1922 3          RESTORE_CONTEXT ();
: 935      1923 3          USER_STATUS = .SAVE_STATUS;
: 936      1924 3          READ_HEADER (CURRENT_FIB[FIB$W_FID], .PRIMARY_FCB);
: 937      1925 2          END;
: 938      1926 2
: 939      1927 2      ! If we make it this far, it means that: 1) there was no previous version of
```



CREATE  
V04-001

L 1  
16-Sep-1984 00:06:06  
14-Sep-1984 12:30:13

VAX-11 Bliss-32 V4.0-742  
DISK\$VMSMASTER:[F11X.SRC]CREATE.B32;2  
Page 26  
(3)

```
: 940      1928 2 ! the file; 2) the previous version of the file is not accessible; or 3) the
: 941      1929 2 ! current process does not have access to the previous version. In any of
: 942      1930 2 ! these cases, propagate as a newly created file.
: 943      1931 2
: 944      1932 2 STATUS = KERNEL_CALL (COPY_INFO, .DIR_FCB, .PRIMARY_FCB, .FIB, 1);
: 945      1933 2
: 946      1934 2 RETURN .STATUS;
: 947      1935 2
: 948      1936 1 END;
```

! End of routine PROPAGATE\_ATTR

.EXTRN OPEN\_FILE, CLOSE\_FILE

007C 00000 PROPAGATE\_ATTR:

	55	08	AA	9E	00002	.WORD	Save R2,R3,R4,R5,R6	1814
	54	014C	CA	9E	00006	MOVAB	8(BASE), R5	1865
	6D	00BF	CF	DE	0000B	MOVAB	332(BASE), R4	
	50		65	D0	00010	MOVAL	7\$, (FP)	
24	A0		06	29	00013	MOVL	(R5), R0	1888
	64		04	12	00018	CMPC3	#6, (R4), 36(R0)	
			01	D0	0001A	BNEQ	1\$	
	50		04	0001D	MOVL	#1, R0		1889
			64	B5	0001E	RET		
			08	12	00020	TSTW	(R4)	1891
		05	A4	95	00022	BNEQ	2\$	
			03	12	00025	TSTB	5(R4)	1892
			008D	31	00027	BNEQ	2\$	
			AA	D0	0002A	BRW	5\$	
CO	AA	80	AA	D0	0002A	MOVL	-128(BASE), -64(BASE)	1896
	56		65	D0	0002F	MOVL	(R5), FILE_FCB	1897
	53	14	AA	D0	00032	MOVL	20(BASE), SAVCURRINDX	1898
0000G	CF		00	FB	00036	CALLS	#0, SAVE_CONTEXT	1899
			02	DD	0003B	PUSHL	#2	1900
			54	DD	0003D	PUSHL	R4	
0000G	CF		02	FB	0003F	CALLS	#2, OPEN_FILE	
	52		50	D0	00044	MOVL	R0, WINDOW	
			58	13	00047	BEQL	4\$	1901
	54	18	A2	D0	00049	MOVL	24(WINDOW), FCB	1904
	51	90	AA	D0	0004D	MOVL	-112(BASE), R1	1906
	50	04	AC	D0	00051	MOVL	FIB, R0	
7E	0B	A1	00	EF	00055	EXTZV	#0, #2, 11(R1), -(SP)	
	02		A0	91	0005B	CMPB	46(R0), (SP)	
	6E	2E	04	1B	0005F	BLEQU	3\$	
	6E	2E	A0	9A	00061	MOVZBL	46(R0), (SP)	
			65	DD	00065	PUSHL	(R5)	1905
	7E		04	7D	00067	MOVQ	#4, -(SP)	
0000G	CF		04	FB	0006A	CALLS	#4, CHECK_PROTECT	
	2F		50	E9	0006F	BLBC	R0, 4\$	
14	AA		53	D0	00072	MOVL	SAVCURRINDX, 20(BASE)	1914
			7E	D4	00076	CLRL	-(SP)	1915
		04	AC	DD	00078	PUSHL	FIB	
		0050	8F	BB	0007B	PUSHR	#4M<R4,R6>	
0000V	CF		04	FB	0007F	CALLS	#4, COPY_INFO	
	53		50	D0	00084	MOVL	R0, STATUS	
			52	DD	00087	PUSHL	WINDOW	1916
0000G	CF		01	FB	00089	CALLS	#1, CLOSE_FILE	



CREATE  
V04-001

M 1  
16-Sep-1984 00:06:06  
14-Sep-1984 12:30:13

VAX-11 Bliss-32 V4.0-742  
DISK\$VMSMASTER:[F11X.SRC]CREATE.B32;2

Page 27  
(3)

	0000G	CF		00	FB	0008E	CALLS	#0, RESTORE_CONTEXT	:	1917
				65	DD	00093	PUSHL	(R5)	:	1918
7E	10	AA		04	C1	00095	ADDL3	#4, 16(BASE), -(SP)	:	
	0000G	CF		02	FB	0009A	CALLS	#2, READ_HEADER	:	
				29	11	0009F	BRB	6\$	:	1919
	0000G	CF		00	FB	000A1	CALLS	#0, RESTORE_CONTEXT	:	1922
	80	AA	C0	AA	D0	000A6	MOVL	-64(BASE), -128(BASE)	:	1923
				65	DD	000AB	PUSHL	(R5)	:	1924
7E	10	AA		04	C1	000AD	ADDL3	#4, 16(BASE), -(SP)	:	
	0000G	CF		02	FB	000B2	CALLS	#2, READ_HEADER	:	
				01	DD	000B7	PUSHL	#1	:	1932
			04	AC	DD	000B9	PUSHL	FIB	:	
				65	DD	000BC	PUSHL	(R5)	:	
			00D0	CA	DD	000BE	PUSHL	208(BASE)	:	
	0000V	CF		04	FB	000C2	CALLS	#4, COPY_INFO	:	
		53		50	D0	000C7	MOVL	R0, STATUS	:	
		50		53	D0	000CA	MOVL	STATUS, R0	:	1934
				04	000CD	RET			:	1936
				0000	000CE	7\$:	.WORD	Save nothing	:	1865
				7E	D4	000D0	CLRL	-(SP)	:	
				5E	DD	000D2	PUSHL	SP	:	
		7E	04	AC	7D	000D4	MOVQ	4(AP), -(SP)	:	
	0000V	CF		03	FB	0C0D8	CALLS	#3, PROPAGATE_HANDLER	:	
				04	000DD	RET			:	

; Routine Size: 222 bytes, Routine Base: \$CODE\$ + 058F



```

: 950      1937 1 ROUTINE PROPAGATE_HANDLER (SIGNAL, MECHANISM) =
: 951      1938 1
: 952      1939 1 ++
: 953      1940 1
: 954      1941 1 FUNCTIONAL DESCRIPTION:
: 955      1942 1
: 956      1943 1     This routine is the condition handler for the file attribute
: 957      1944 1     propagation. It unwinds and returns a value of zero to
: 958      1945 1     indicate a failure.
: 959      1946 1
: 960      1947 1 CALLING SEQUENCE:
: 961      1948 1     PROPAGATE_HANDLER (ARG1, ARG2)
: 962      1949 1
: 963      1950 1 INPUT PARAMETERS:
: 964      1951 1     ARG1: address of the signal array
: 965      1952 1     ARG2: address of the mechanism array
: 966      1953 1
: 967      1954 1 IMPLICIT INPUTS:
: 968      1955 1     none
: 969      1956 1
: 970      1957 1 OUTPUT PARAMETERS:
: 971      1958 1     none
: 972      1959 1
: 973      1960 1 IMPLICIT OUTPUTS:
: 974      1961 1     Value of the routine that caused the exception is returned as zero.
: 975      1962 1
: 976      1963 1 ROUTINE VALUE:
: 977      1964 1     SS$_RESIGNAL or none
: 978      1965 1
: 979      1966 1 SIDE EFFECTS:
: 980      1967 1     none
: 981      1968 1
: 982      1969 1 --
: 983      1970 1
: 984      1971 2 BEGIN
: 985      1972 2
: 986      1973 2 MAP
: 987      1974 2     SIGNAL          : REF BBLOCK,          ! Signal argument array
: 988      1975 2     MECHANISM      : REF BBLOCK;          ! Mechanism argument array
: 989      1976 2
: 990      1977 2 ! If the condition is change mode to user (ERR_EXIT) set the saved value
: 991      1978 2 ! of R0 to zero (indicating a failure) and unwind to the PROPAGATE_ATTR
: 992      1979 2 ! routine.
: 993      1980 2
: 994      1981 2 IF .SIGNAL[CHFS$_SIG_NAME] EQL SS$_CMODUSER
: 995      1982 2 THEN
: 996      1983 3     BEGIN
: 997      1984 3         MECHANISM[CHFS$_MCH_SAVRO] = 0;          ! Note failure
: 998      1985 3         $UNWIND (DEPADR = MECHANISM[CHFS$_MCH_DEPTH],
: 999      1986 3             NEWPC = 0);
: 1000     1987 2     END;
: 1001     1988 2
: 1002     1989 2 RETURN SS$_RESIGNAL;          ! Ignored when unwinding
: 1003     1990 2
: 1004     1991 1 END;          ! End of routine PROPAGATE_HANDLER

```



CREATE  
V04-001

B 2  
16-Sep-1984 00:06:06  
14-Sep-1984 12:30:13

VAX-11 Bliss-32 V4.0-742  
DISK\$VMSMASTER:[F11X.SRC]CREATE.B32;2  
Page 29  
(4)

.EXTRN SYSSUNWIND

				0000 00000 PROPAGATE HANDLER:					
		50	04	AC	D0	00002	.WORD	Save nothing	: 1937
00000424		8F	04	A0	D1	00006	MOVL	SIGNAL, R0	: 1981
				15	12	0000E	CMPL	4(R0), #1060	: :
		50	08	AC	D0	00010	BNEQ	1\$	: :
			0C	A0	D4	00014	MOVL	MECHANISM, R0	: 1984
				7E	D4	00017	CLRL	12(R0)	: :
7E	08	AC		08	C1	00019	CLRL	-(SP)	: 1986
				02	FB	0001E	ADDL3	#8, MECHANISM, -(SP)	: :
00000000G		00		02	FB	0001E	CALLS	#2, SYSSUNWIND	: :
		50	0918	8F	3C	00025	MOVZWL	#2328, R0	: 1989
				04	0002A	1\$:	RET		: 1991

; Routine Size: 43 bytes,      Routine Base: \$CODE\$ + 066D



```
1006 1992 1 ROUTINE COPY_INFO (OLD_FILE_FCB, NEW_FILE_FCB, FIB, NEW_FILE) : L_NORM =
1007 1993 1
1008 1994 1 ++
1009 1995 1
1010 1996 1 FUNCTIONAL DESCRIPTION:
1011 1997 1
1012 1998 1 This routine actually copies the propagated information. This
1013 1999 1 routine must be called in kernel mode. The propagation takes
1014 2000 1 place according to the following rules:
1015 2001 1
1016 2002 1 UIC - For a newly created file, the file takes the UIC of the
1017 2003 1 creator unless the creator has resource rights to the
1018 2004 1 owner of the directory. In which case, the UIC of the
1019 2005 1 directory owner is used. For a new version of an
1020 2006 1 existing file, the UIC of the creator is used if the
1021 2007 1 creator does not have resource rights to either the
1022 2008 1 old version owner or the directory owner. If the
1023 2009 1 creator has resource rights to the old version owner,
1024 2010 1 that UIC is used. If not, and the creator has resource
1025 2011 1 rights to the directory owner, the directory owner
1026 2012 1 UIC is used.
1027 2013 1
1028 2014 1 Protection - For a newly created file, the protection is taken from
1029 2015 1 the directory default protection ACE, if it exists. If
1030 2016 1 it does not exist, the process default protection is used.
1031 2017 1 For a new version of an existing file, the protection is
1032 2018 1 taken from the old version of the file.
1033 2019 1
1034 2020 1 ACL - For a newly created file, the ACL is taken from the
1035 2021 1 directory default ACL. If no directory default ACL
1036 2022 1 exists, no ACL is propagated. For a new version of
1037 2023 1 an existing file, the ACL is taken from the old
1038 2024 1 version of the file.
1039 2025 1
1040 2026 1 CALLING SEQUENCE:
1041 2027 1 COPY_INFO (ARG1, ARG2, ARG3, ARG4)
1042 2028 1
1043 2029 1 INPUT PARAMETERS:
1044 2030 1 ARG1: address of the old file's FCB (if one)
1045 2031 1 ARG2: address of the new file's FCB
1046 2032 1 ARG3: address of the FIB
1047 2033 1 ARG4: 1 if defaults for a new file
1048 2034 1 0 if defaults for a new version of an existing file
1049 2035 1
1050 2036 1 IMPLICIT INPUTS:
1051 2037 1 DIR_FCB: address of parent directory FCB
1052 2038 1
1053 2039 1 OUTPUT PARAMETERS:
1054 2040 1 none
1055 2041 1
1056 2042 1 IMPLICIT OUTPUTS:
1057 2043 1 none
1058 2044 1
1059 2045 1 ROUTINE VALUE:
1060 2046 1 1
1061 2047 1
1062 2048 1 SIDE EFFECTS:
```



```
1063 2049 1 | The ACL building routine is called to update the new file's file
1064 2050 1 | headers with the copied ACL.
1065 2051 1 |
1066 2052 1 | --
1067 2053 1 |
1068 2054 2 BEGIN
1069 2055 2
1070 2056 2 MAP
1071 2057 2     OLD_FILE_FCB : REF BBLOCK,      ! Address of old file's FCB
1072 2058 2     NEW_FILE_FCB : REF BBLOCK,    ! Address of new file's FCB
1073 2059 2     FIB : REF BBLOCK;             ! Address of the FIB
1074 2060 2
1075 2061 2 LINKAGE
1076 2062 2     L_SEARCH_RIGHT = JSB (REGISTER = 2, REGISTER = 4;
1077 2063 2                     REGISTER = 1, REGISTER = 5);
1078 2064 2
1079 2065 2     L_FINDACL = JSB (REGISTER = 3, REGISTER = 5,
1080 2066 2                     REGISTER = 6, REGISTER = 1;
1081 2067 2                     REGISTER = 1);
1082 2068 2
1083 2069 2 LOCAL
1084 2070 2     PCB : REF BBLOCK,      ! PCB address of I/O packet owner
1085 2071 2     ARB : REF BBLOCK,     ! Access rights block address
1086 2072 2     IDENTIFIER,          ! Identifier being sought
1087 2073 2     RIGHTS_DESC,         ! Rights list descr addr
1088 2074 2     ID_FOUND : REF BBLOCK, ! Addr of ID found
1089 2075 2     RIGHTS_SEG : REF BBLOCK, ! Addr of rights segment
1090 2076 2     ACE_ADDRESS : REF BBLOCK, ! Pointer to default protection ACE
1091 2077 2     OLD_ACL_SEGMENT : REF BBLOCK, ! Address of old ACL segment
1092 2078 2     NEW_ACL_SEGMENT : REF BBLOCK; ! Address of new ACL segment
1093 2079 2
1094 2080 2 EXTERNAL
1095 2081 2     SCH$GL_PCBVEC : REF VECTOR ADDRESSING_MODE (ABSOLUTE); ! PCB vector
1096 2082 2
1097 2083 2 BIND_COMMON;
1098 2084 2
1099 2085 2 EXTERNAL ROUTINE
1100 2086 2     EX$SEARCH_RIGHT : L_SEARCH_RIGHT ADDRESSING_MODE (GENERAL),
1101 2087 2                     ! Search for specified ID
1102 2088 2     EX$FINDACL : L_FINDACL ADDRESSING_MODE (GENERAL), ! Locate an ACE
1103 2089 2     ACL_INIT_QUEUE : ADDRESSING_MODE (GENERAL), ! Initialize ACL queue
1104 2090 2     ACL_COPYACL : L_NORM, ! Routine to propagate desired ACEs
1105 2091 2     CHANGE_OWNER : L_NORM; ! Change file owner UIC
1106 2092 2
1107 2093 2 ENABLE PROPAGATE_HANDLER;
1108 2094 2
1109 2095 2 ! Initialize some necessary pointers.
1110 2096 2
1111 2097 2 PCB = .SCH$GL_PCBVEC[(IO_PACKET[IRP$L_PID])<0,16>];
1112 2098 2 ARB = .IO_PACKET[IRP$L_ARB];
1113 2099 2 RIGHTS_DESC = ARB[ARB$RIGHTSLIST];
1114 2100 2
1115 2101 2 ! If is a new file, propagate the information from the parent directory
1116 2102 2 ! or the creator of the file as necessary.
1117 2103 2
1118 2104 2 IF .NEW_FILE
1119 2105 2 THEN
```



```
1120 2106 3 BEGIN
1121 2107 3 IF .DIR_FCB NEQ 0
1122 2108 3 THEN
1123 2109 4 BEGIN
1124 2110 4 CHANGE_OWNER (.DIR_FCB[FCB$L_FILEOWNER], .NEW_FILE_FCB, 0);
1125 2111 4 NEW_FILE_FCB[FCB$W_FILEPROT] = .PCB[PCB$L_DEFPROT];
1126 2112 4 IF .BBLOCK[DIR_FCB[FCB$R_ORB], ORB$V_ACL_QUEUE]
1127 2113 4 THEN
1128 2114 5 BEGIN
1129 2115 5 OLD_ACL_SEGMENT = .DIR_FCB[FCB$L_ACLFL];
1130 2116 5 UNTIL .OLD_ACL_SEGMENT EQ LA DIR_FCB[FCB$L_ACLFL]
1131 2117 5 DO
1132 2118 6 BEGIN
1133 2119 6 ACE_ADDRESS = 0;
1134 2120 6 IF EXES$FINDACL (ACES$DIRDEF,
1135 2121 6 .OLD_ACL_SEGMENT[ACL$W_SIZE] - ACL$C_LENGTH,
1136 2122 6 OLD_ACL_SEGMENT[ACL$L_LIST], .ACE_ADDRESS;
1137 2123 6 ACE_ADDRESS)
1138 2124 6 THEN
1139 2125 7 BEGIN
1140 2126 7 (NEW_FILE_FCB[FCB$W_FILEPROT])<0,4> = .ACE_ADDRESS[ACES$L_SYS_PROT];
1141 2127 7 (NEW_FILE_FCB[FCB$W_FILEPROT])<4,4> = .ACE_ADDRESS[ACES$L_OWN_PROT];
1142 2128 7 (NEW_FILE_FCB[FCB$W_FILEPROT])<8,4> = .ACE_ADDRESS[ACES$L_GRP_PROT];
1143 2129 7 (NEW_FILE_FCB[FCB$W_FILEPROT])<12,4> = .ACE_ADDRESS[ACES$L_WOR_PROT];
1144 2130 7 EXIT[LOOP];
1145 2131 6 END;
1146 2132 6 OLD_ACL_SEGMENT = .OLD_ACL_SEGMENT[ACL$L_FLINK];
1147 2133 5 END;
1148 2134 5 ACL_INIT_QUEUE (NEW_FILE_FCB[FCB$R_ORB]);
1149 2135 6 RETURN ACL_COPYACL ? .DIR_FCB, .NEW_FILE_FCB, (IF .FIB[FIB$V_DIRACL]
1150 2136 5 THEN 2 ELSE 1));
1151 2137 4 END;
1152 2138 4 RETURN 1;
1153 2139 3 END;
1154 2140 2 END;
1155 2141 2
1156 2142 2 ! If it is a new version of an existing file, propagate the information
1157 2143 2 ! from the old version of the file, the parent directory, or the creator
1158 2144 2 ! of the file.
1159 2145 2
1160 2146 2 ! First, set the owner of the new file.
1161 2147 2
1162 2148 2 IF NOT CHANGE_OWNER (.OLD_FILE_FCB[FCB$L_FILEOWNER], .NEW_FILE_FCB, 0)
1163 2149 2 AND .DIR_FCB NEQ 0
1164 2150 2 THEN CHANGE_OWNER (.DIR_FCB[FCB$L_FILEOWNER], .NEW_FILE_FCB, 0);
1165 2151 2
1166 2152 2 ! Next, propagate the protection from the old file.
1167 2153 2
1168 2154 2 NEW_FILE_FCB[FCB$W_FILEPROT] = .OLD_FILE_FCB[FCB$W_FILEPROT];
1169 2155 2
1170 2156 2 ! Last, but not least, copy the ACL (excluding ACEs marked as NOPROPAGATE).
1171 2157 2
1172 2158 2 IF .BBLOCK[OLD_FILE_FCB[FCB$R_ORB], ORB$V_ACL_QUEUE]
1173 2159 2 THEN
1174 2160 3 BEGIN
1175 2161 3 ACL_INIT_QUEUE (NEW_FILE_FCB[FCB$R_ORB]);
1176 2162 3 RETURN ACL_COPYACL ? .OLD_FILE_FCB, .NEW_FILE_FCB, 2)
```



```

: 1177      2163  3      END
: 1178      2164  2      ELSE RETURN 1;
: 1179      2165  2
: 1180      2166  1      END;

```

```
! End of routine COPY_INFO
```

```
.EXTRN  EX$SEARCH_RIGHT
.EXTRN  EX$FINDACL, ACL_COPYACL
.EXTRN  CHANGE_OWNER
```

OBFC 00000 COPY\_INFO:

Address	Hex	Op	Op2	Op3	Op4	Op5	Op6	Op7	Op8	Op9	Op10	Op11	Op12	Op13	Op14	Op15	Op16	Op17	Op18	Op19	Op20	Op21	Op22	Op23	Op24	Op25	Op26	Op27	Op28	Op29	Op30	Op31	Op32	Op33	Op34	Op35	Op36	Op37	Op38	Op39	Op40	Op41	Op42	Op43	Op44	Op45	Op46	Op47	Op48	Op49	Op50	Op51	Op52	Op53	Op54	Op55	Op56	Op57	Op58	Op59	Op60	Op61	Op62	Op63	Op64	Op65	Op66	Op67	Op68	Op69	Op70	Op71	Op72	Op73	Op74	Op75	Op76	Op77	Op78	Op79	Op80	Op81	Op82	Op83	Op84	Op85	Op86	Op87	Op88	Op89	Op90	Op91	Op92	Op93	Op94	Op95	Op96	Op97	Op98	Op99	Op100	Op101	Op102	Op103	Op104	Op105	Op106	Op107	Op108	Op109	Op110	Op111	Op112	Op113	Op114	Op115	Op116	Op117	Op118	Op119	Op120	Op121	Op122	Op123	Op124	Op125	Op126	Op127	Op128	Op129	Op130	Op131	Op132	Op133	Op134	Op135	Op136	Op137	Op138	Op139	Op140	Op141	Op142	Op143	Op144	Op145	Op146	Op147	Op148	Op149	Op150	Op151	Op152	Op153	Op154	Op155	Op156	Op157	Op158	Op159	Op160	Op161	Op162	Op163	Op164	Op165	Op166	Op167	Op168	Op169	Op170	Op171	Op172	Op173	Op174	Op175	Op176	Op177	Op178	Op179	Op180	Op181	Op182	Op183	Op184	Op185	Op186	Op187	Op188	Op189	Op190	Op191	Op192	Op193	Op194	Op195	Op196	Op197	Op198	Op199	Op200	Op201	Op202	Op203	Op204	Op205	Op206	Op207	Op208	Op209	Op210	Op211	Op212	Op213	Op214	Op215	Op216	Op217	Op218	Op219	Op220	Op221	Op222	Op223	Op224	Op225	Op226	Op227	Op228	Op229	Op230	Op231	Op232	Op233	Op234	Op235	Op236	Op237	Op238	Op239	Op240	Op241	Op242	Op243	Op244	Op245	Op246	Op247	Op248	Op249	Op250	Op251	Op252	Op253	Op254	Op255	Op256	Op257	Op258	Op259	Op260	Op261	Op262	Op263	Op264	Op265	Op266	Op267	Op268	Op269	Op270	Op271	Op272	Op273	Op274	Op275	Op276	Op277	Op278	Op279	Op280	Op281	Op282	Op283	Op284	Op285	Op286	Op287	Op288	Op289	Op290	Op291	Op292	Op293	Op294	Op295	Op296	Op297	Op298	Op299	Op300	Op301	Op302	Op303	Op304	Op305	Op306	Op307	Op308	Op309	Op310	Op311	Op312	Op313	Op314	Op315	Op316	Op317	Op318	Op319	Op320	Op321	Op322	Op323	Op324	Op325	Op326	Op327	Op328	Op329	Op330	Op331	Op332	Op333	Op334	Op335	Op336	Op337	Op338	Op339	Op340	Op341	Op342	Op343	Op344	Op345	Op346	Op347	Op348	Op349	Op350	Op351	Op352	Op353	Op354	Op355	Op356	Op357	Op358	Op359	Op360	Op361	Op362	Op363	Op364	Op365	Op366	Op367	Op368	Op369	Op370	Op371	Op372	Op373	Op374	Op375	Op376	Op377	Op378	Op379	Op380	Op381	Op382	Op383	Op384	Op385	Op386	Op387	Op388	Op389	Op390	Op391	Op392	Op393	Op394	Op395	Op396	Op397	Op398	Op399	Op400	Op401	Op402	Op403	Op404	Op405	Op406	Op407	Op408	Op409	Op410	Op411	Op412	Op413	Op414	Op415	Op416	Op417	Op418
---------	-----	----	-----	-----	-----	-----	-----	-----	-----	-----	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------



CREATE  
V04-001

G 2  
16-Sep-1984 00:06:06  
14-Sep-1984 12:30:13

VAX-11 Bliss-32 V4.0-742  
DISK\$VMSMASTER:[F11X.SRC]CREATE.B32;2

Page 34  
(5)

		52		05	11	000BB		BRB	6\$		2125
				62	D0	000BD	5\$:	MOVL	(OLD_ACL_SEGMENT), OLD_ACL_SEGMENT		2132
				A7	11	000C0		BRB	4\$		2116
7E	08	AC	00000058	8F	C1	000C2	6\$:	ADDL3	#88, NEW_FILE_FCB, -(SP)		2134
		67		01	FB	000CB		CALLS	#1, ACL_INIT_QUEUE		
		50	0C	AC	D0	000CE		MOVL	FIB, R0		2135
04	38	A0		02	E1	000D2		BBC	#2, 56(R0), 7\$		
				02	DD	000D7		PUSHL	#2		
				02	11	000D9		BRB	8\$		
				01	DD	000DB	7\$:	PUSHL	#1		
			08	AC	DD	000DD	8\$:	PUSHL	NEW_FILE_FCB		
				64	DD	000E0		PUSHL	(R4)		
				48	11	000E2		BRB	11\$		
				7E	D4	000E4	9\$:	CLRL	-(SP)		2148
			08	AC	DD	000E6		PUSHL	NEW_FILE_FCB		
		50		AC	D0	000E9		MOVL	OLD_FILE_FCB, R0		
			04	A0	DD	000ED		PUSHL	88(R0)		
		68		03	FB	000F0		CALLS	#3, CHANGE_OWNER		
		12		50	E8	000F3		BLBS	R0, 10\$		
				64	D5	000F6		TSTL	(R4)		2149
				0E	13	000F8		BEQL	10\$		
				7E	D4	000FA		CLRL	-(SP)		2150
			08	AC	DD	000FC		PUSHL	NEW_FILE_FCB		
		50		64	D0	000FF		MOVL	(R4), R0		
			58	A0	DD	00102		PUSHL	88(R0)		
		68		03	FB	00105		CALLS	#3, CHANGE_OWNER		
		50	04	AC	7D	00108	10\$:	MOVQ	OLD_FILE_FCB, R0		2154
	70	A1	70	A0	B0	0010C		MOVW	112(R0), 112(R1)		
		50	04	AC	D0	00111		MOVL	OLD_FILE_FCB, R0		2158
18		A0		01	E1	00115		BBC	#1, 99(R0), 12\$		
7E	63	AC	00000058	8F	C1	0011A		ADDL3	#88, NEW_FILE_FCB, -(SP)		2161
		67		01	FB	00123		CALLS	#1, ACL_INIT_QUEUE		
				02	DD	00126		PUSHL	#2		2162
		7E	04	AC	7D	00128		MOVQ	OLD_FILE_FCB, -(SP)		
	0000G	CF		03	FB	0012C	11\$:	CALLS	#3, ACL_COPYACL		
					04	00131		RET			2164
		50		01	D0	00132	12\$:	MOVL	#1, R0		
					04	00135		RET			2166
					0000	00136	13\$:	.WORD	Save nothing		2081
				7E	D4	00138		CLRL	-(SP)		
				5E	DD	0013A		PUSHL	SP		
		7E	04	AC	7D	0013C		MOVQ	4(AP), -(SP)		
	FE90	CF		03	FB	00140		CALLS	#3, PROPAGATE_HANDLER		
				04	00145			RET			

; Routine Size: 326 bytes, Routine Base: \$CODE\$ + 0698

; 1181 2167 1  
; 1182 2168 1 END  
; 1183 2169 0 ELUDOM



CREATE  
V04-001

H 2  
16-Sep-1984 00:06:06  
14-Sep-1984 12:30:13

VAX-11 Bliss-32 V4.0-742  
DISK\$VMSMASTER:[F11X.SRC]CREATE.B32;2  
Page 35  
(5)

PSECT SUMMARY

Name	Bytes	Attributes
\$CODE\$	2014	NOVEC,NOWRT, RD , EXE,NOSHR, LCL, REL, CON,NOPI,ALIGN(2)

Library Statistics

File	----- Total	Symbols Loaded	----- Percent	Pages Mapped	Processing Time
_\$255\$DUA28:[SYSLIB]LIB.L32;1	18619	140	0	1000	00:01.9

COMMAND QUALIFIERS

BLISS/CHECK=(FIELD,INITIAL,OPTIMIZE)/LIS=LIS\$:CREATE/OBJ=OBJ\$:CREATE MSRC\$:CREATE/UPDATE=(ENH\$:CREATE)

; Size: 2014 code + 0 data bytes  
; Run Time: 01:09.9  
; Elapsed Time: 02:20.0  
; Lines/CPU Min: 1862  
; Lexemes/CPU-Min: 37116  
; Memory Used: 549 pages  
; Compilation Complete



0168

DIGITAL EQUIPMENT CORPORATION  
CONFIDENTIAL AND PROPRIETARY



0169 AH-BT13A-SE  
VAX/VMS V4.0

DIGITAL EQUIPMENT CORPORATION  
CONFIDENTIAL AND PROPRIETARY

